Helsinki University of Technology Automation Technology
Series A: Research Reports No. 33
Espoo, June 2009

A SENSOR-BASED PERSONAL NAVIGATION SYSTEM AND ITS APPLICATION FOR INCORPORATING HUMANS INTO A HUMAN-ROBOT TEAM

Jari Saarinen

TEKNILLINEN KORKEAKOULU
TEKNISKA HÖGSKOLAN
HELSINKI UNIVERSITY OF TECHNOLOGY
TECHNISCHE UNIVERSITÄT HELSINKI
UNIVERSITE DE TECHNOLOGIE D'HELSINKI

Helsinki University of Technology
Automation Technology
Series A: Research Reports No. 33
Espoo, June 2009

A Sensor-Based Personal Navigation System and Its Application for Incorporating Humans Into a Human-Robot Team

Jari Saarinen

Dissertation for the degree of Doctor of Science in Technology to be presented with due permission of the Faculty of Electronics, Communications and Automation for public examination and debate in Auditorium AS1 at Helsinki University of Technology (Espoo, Finland) on the 25th of June, 2009, at 12 noon.

Helsinki University of Technology
Faculty of Electronics, Communications and Automation
Departement of Automation and Systems Technology

Distribution:

Helsinki University of Technology
Faculty of Electronics, Communications and Automation
Departement of Automation and Systems Technology
P.O. Box 5500
FIN-02015 TKK
FINLAND

e-mail: Jari.Saarinen@tkk.fi
Tel. +358 9 451 5146
Fax +358 9 451 3308

ISBN 978-951-22-9961-4 (print)
ISBN 978-951-22-9962-1 (pdf)
ISSN 0783-5477

Yliopistopaino
Helsinki 2009

Available on net at http://lib.tkk.fi/Diss/2009/isbn9789512299621

HELSINKI UNIVERSITY OF TECHNOLOGY P.O. BOX 1000, FI-02150 TKK http://www.tkk.fi	ABSTRACT OF DOCTORAL DISSERTATION

Author: Jari Saarinen

Name of the dissertation:
A Sensor-Based Personal Navigation System and Its Application for Incorporating Humans Into a Human-Robot Team

Date of manuscript: 6.3.2009	Date of the dissertation: 25.6.2009

Type of manuscript: Monograph

Department: Automation and Systems Technology
Research group: Centre of Excellence in Generic Intelligent Machines Research
Field of Research: Mobile robotics
Opponent: Professor Hubert Roth
Supervisor: Professor Aarne Halme

Abstract

In this thesis methods for the sensor-based localisation of human beings are studied. The thesis presents the theory, test results and a realisation of the methods, which is called PeNa. PeNa is further applied to incorporate a human into a human-robot team that performs a simulated search and rescue task.

Human-robot teamwork provides the vision for this thesis. Furthermore, the PeLoTe project and its search and rescue task provided the primary motivation for the research. However, the major part of this work and contribution is on sensor-based personal navigation.

The approaches studied for personal navigation systems are based on sensor-based dead reckoning, laser-based dead reckoning, and map-based localisation. Sensor-based dead reckoning is based on heading estimation using a compass and gyro and step length estimation. Two alternative step length estimation methods are presented, ultrasound-based and accelerometer-based. Two laser dead reckoning methods are presented; a pose correlation method and a combined angle histogram matcher with position correlation. Furthermore, there are three variations for map-based localisation based on the well-known Monte Carlo Localisation (MCL): topological MCL, scan-based MCL, and a combined MCL method.

As a result of the research it can be stated that it is possible to build a personal navigation system that can localise a human being indoors using only self-contained sensors. The results also show that this can be achieved using various combinations of sensors and methods. Furthermore, the personal navigation system that was developed is used to incorporate a human being into a human-robot team performing a search and rescue task. The initial results show that the location information provides a basis for creating situational awareness for a spatially distributed team.

Keywords: Sensor-based personal navigation, map-based indoor localisation, human-robot team

ISBN (printed): 978-951-22-9961-4	ISSN (printed): 0783-5477
ISBN (pdf): 978-951-22-9962-1	ISSN (pdf):
ISBN (others):	Number of pages: 191

Publisher: Helsinki University of Technology, Department of Automation and Systems Technology

Print distribution: Helsinki University of Technology, Department of Automation and Systems Technology

The dissertation can be read at http://lib.tkk.fi/Diss

TEKNILLINEN KORKEAKOULU PL 1000, 02015 TKK http://www.tkk.fi	VÄITÖSKIRJAN TIIVISTELMÄ

Tekijä: Jari Saarinen

Väitöskirjan nimi:
Anturipohjainen henkilökohtainen navigointijärjestelmä ja sen soveltaminen ihmisten liittämiseksi ihminen-robotti joukkueeseen

Käsikirjoituksen jättöpäivä: 6.3.2009	Väitöstilaisuuden ajankohta: 25.6.2009

Käsikirjoituksen tyyppi: Monografia

Laitos: Automaatio- ja Systeemitekniikan laitos
Tutkimusryhmä: Älykkäiden koneiden huippuyksikkö
Tutkimusala: Liikkuvat robotit
Vastaväittäjä: Professori Hubert Roth
Valvoja ja ohjaaja: Professori Aarne Halme

Tiivistelmä
Tässä väitöskirjassa tutkitaan menetelmiä ihmisen anturipohjaiseen paikantamiseen. Työssä esitetään menetelmien teoria, testitulokset ja käytännön toteutus laitteistolla, jota kutsutaan PeNaksi. PeNa laitteistoa sovelletaan lisäksi ihmisen liittämiseksi ihminen-robotti joukkueeseen, joka suorittaa simuloitua pelastustehtävää.
Ihminen-robotti yhteistyö tarjoaa työlle vision. Lisäksi PeLoTe projekti ja sen pelastustehtävä demonstraatio ovat toimineet työlle motivaationa, mutta työn painopiste ja kontribuutio ovat anturipohjaisen henkilökohtaisen navigoinnin kehittämisessä.
Työssä tutkitut menetelmät perustuvat anturipohjaisiin merkintälaskumenetelmiin, laser-pohjaisiin merkintälaskumenetelmiin, ja kartta-pohjaisiin paikannusmenetelmiin. Anturipohjainen merkintälasku perustuu kulman estimointiin käyttäen kompassia ja gyroskooppia, sekä askelpituuden estimointiin. Työssä esitellään kaksi vaihtoehtoista askelpituudenestimointimenetelmää, ultraäänipohjainen ja kiihtyvyysanturipohjainen.
Työssä tutkitaan myös laserin käyttöä henkilökohtaiseen navigointiin ja siinä esitellään kaksi eri menetelmää laserpohjaiseen merkintälaskuun: asento-avaruus korrelaatio menetelmä, ja yhdistetty kulmahistogrammikorrelaatiomenetelmä yhdistettynä paikka pohjaiseen korrelaatiomenetelmään. Lisäksi työssä esitetään kolme eri muunnelmaa karttapohjaiseen paikannukseen, jotka perustuvat tunnettuun Monte Carlo paikannusmenetelmään.
Työn tuloksena voidaan todeta, että ihmisen anturipohjainen sisätilapaikannus on mahdollista. Tulokset osoittavat myös, että tämä on mahdollista saavuttaa käyttäen eri anturiyhdistelmiä.
Lisäksi työssä kehitettyä järjestelmää sovelletaan ihmisen liittämiseksi ihminen-robotti järjestelmään, joka suorittaa pelastustehtävää. Testien alustavat tulokset osoittavat, että paikkatietoa voidaan hyödyntää tilannetietoisuuden luomisessa spatiaalisesti hajautetussa järjestelmässä.

Avainsanat: Anturipohjainen henkilökohtainen navigointi, karttapohjainen sisätilapaikannus, ihminen-robotti joukkue

ISBN (painettu): 978-951-22-9961-4	ISSN (painettu): 0783-5477
ISBN (pdf): 978-951-22-9962-1	ISSN (pdf):
ISBN (muut):	Sivumäärä: 191

Julkaisija: Teknillinen korkeakoulu, Automaatio- ja Systeemitekniikan laitos

Painetun väitöskirjan jakelu: Teknillinen korkeakoulu, Automaatio- ja Systeemitekniikan laitos

Luettavissa verkossa osoitteessa http://lib.tkk.fi/Diss

Preface

This study was conducted at the Automation Technology Laboratory of Helsinki University of Technology during the years 2002-2008. The major part of the work was conducted within a research project called PeLoTe. PeLoTe is an abbreviation from "Building Presence through Localisation for Hybrid Telematic Systems" and it was a European Union-funded project, IST-2001-38873. The project was active between 2002 and 2005. The work was also supported by the Finnish Academy of Sciences, the Czech Academy of Sciences, and the MIRACLE project (grant of the European Commission No. ICA1-CT-2000-70002: MIRACLE in the context of the "Centres of Excellence" scheme) during my research exchange in the Czech Republic. Furthermore, the author received personal support from the Eemil Aaltonen Fund. The work was eventually finalised in 2009 at the Centre of Excellence in Generic Intelligent Machines, which is partly funded by the Finnish Academy of Sciences. All the sources of financial support are gratefully acknowledged.

I wish to express my gratitude to Professor Aarne Halme, my supervisor and mentor. Every day in the laboratory for the past ten years has been a joy. Professor Halme has been there for me as a teacher, as a counsellor, and as a source of inspiration. Thank you.

Second, I wish to thank the people who influenced the work that led into this thesis. Docent Mika Vainio and Doctor Jussi Suomela were the ones that wrote the application for the PeLoTe project. Jussi further operated as a project manager at our end. Special thanks go to Seppo Heikkilä and Mikko Elomaa, who were working for me as research assistants and did outstanding work in their areas of responsibility. I would also like to thank the other PeLoTe partners, Dr. Libor Preucil, for being the manager of PeLoTe and giving me the opportunity to work in their laboratory, and Jiri Pavlicek, Roman Mazl, Miroslav Kulich, Frauke Driewer, and Herbert Baier for their valuable assistance and contributions to the project (as well as to the project atmosphere).

I also owe a great debt to many of my co-workers. Jussi Suomela took me into the laboratory and has been my second mentor throughout my career. The endless hours spent discussing and innovating over our cigars also had a significant influence on this thesis. Special thanks also go to Tapio Leppanen, who assisted whenever something needed to be put together, Kalle Rosenblad, who helped me with the electronics, Janne Paanajärvi, who helped me with numerous mathematical issues and Antti Maula, who was dedicated to solving my programming and other infor-

mation technology-related problems. And thanks to all my colleagues who are not mentioned here for providing their contributions to this inspiring community.

Finally, there are no words big enough to express my gratitude towards my family. I thank my father, who has been there for me all these years. He provided me with the security, both financial and mental, required for taking the step and entering the University of Technology in the first place. I thank my wife for being there for my whole career, understanding me when the work truly was my second home, and giving me a loving home with two great sons.

Espoo, June 2009

Jari Saarinen

Contents

Nomenclature

(dx_w, dy_w) Differential movement with respect to world frame of reference.

(r_i^t, θ_i^t) One range, bearing pair of the range scan.

$(x(t), y(t), \varphi(t))$ 2D pose at time t.

α Angle

β Angle

χ_t Pose distribution represented with a set of particles.

$\chi_t^i = (x_t^i, w_t^i)$ One sample, that is, particle of pose distribution.

η Normalisation constant for normalising the posterior distribution after update.

γ Angle

$\hat{x}_t$ Estimate of the state at time t.

$\hat{x}_t^-$ Priori of state estimate at time t.

$(dx_t, dy_t, d\varphi_t)$ Differential movement from pose at time $t-1$ to pose at time t.

μ Mean

ω Natural frequency

$\omega(t)$ Angular rate.

$\omega_b(t)$ Bias term of angular velocity.

π Mathematical pi.

σ Standard deviation.

ζ Damping factor.

A An absolute value of acceleration. Can be used with subscript. For example A_{max} to indicate the maximum value of acceleration.

$a(t)$ Acceleration.

$a_m(t)$ Measured acceleration.

$a_{oe}(t)$ Acceleration caused by orientation error.

$a_d(t)$ Acceleration caused by drift.

C_i Some constant.

$Cor(j)$ Correlation function

d Distance. Can be used with subscripts for clarification. For example d_{nn} marks distance to nearest neighbor.

dl Differential length.

$E(t)$ Transformation matrix for IMU orientation tracking.

e_{x_t} Measurement error of x coordinate. Similarly e_{y_t} is the measurement error of the y-coordinate and e_{phi_t} is the heading measurement error.

$f(\hat{x}_{t-1}, u_t)$ Prototype of a motion model.

g Gravity.

$g(\hat{x}_t^-, z_{t,}m)$ Prototype of measurement model.

$G(s)$ Transfer function

$h(i)$ Histogram function.

J Cost function.

K Kalman gain.

L Likelihood. Can be used with subscripts fo clarification. For example L_r indicates a likelihood of range measurement.

l Length

l_{step} Step length in metres.

l_{stride} Stride length in metres.

M Constant used to indicate the number of particles in distribution.

m Map.

N A constant.

$N(0, \sigma_x^2)$ Normal distribution, with zero mean and $sigma_x^2$ variance.

$O(N)$ Complexity of algorithm.

$p(x_t \mid x_{t-1}, u_t, m)$ State transition probability, that is, motion model.

$p(x_t \mid z_{1:t}, u_{1:t}, m)$ Posterior distribution of pose.

$p(z_t \mid x_t, m)$ Measurement model.

p_t Pose at time t.

Q Prediction variance in Kalman filter.

R Measurement variance of Kalman filter.

R_φ 2D-rotation matrix.

r_m One range measurement. Can be used with different subscripts, e.g. r_p to denote the predicted measurement.

$R_x(\alpha)$ Rotation matrix for roll.

$R_y(\beta)$ Rotation matrix for pitch.

$R_z(\gamma)$ Rotation matrix for yaw.

s_t Range scan taken at time t.

T Translation

t Time index.

u_t Control at time t.

x x-coordinate value.

x_t State at time t.

y y-coordinate value.

z_t Measuremement at time t.

AGV Automatic Guided Vehicle

DOF Degree(s) of Freedom

GPS Global Positioning System

HE Human Entity

HRI Human Robot Interaction

ICP Iterative Closest Point

IDC Iterative Dual Correspondence

IMRP Iterative Matching Range Point

IMU Inertial Measurement Unit

MCL Monte Carlo Localisation

NUPPU Non-Ultrasonic Pedestrian Pedometer Unit

PAS Personal Assistance System

PDR Personal Dead Reckoning

PeLoTe Building Presence through Localisation for Hybrid Telematic Systems. It was a European Union-funded project, IST-2001-38873.

PeNa Self-contained sensor system used for personal navigation testing (abbreviation for Personal Navigation System)

RE Robot Entity

SA Situation Awareness

SiLMU Step Length Measurement Unit

SIR Sampling Importance Resampling

ZUPT Zero-Velocity Update

List of Figures

List of Tables

List of Algorithms

Chapter 1

Introduction

1.1 Background and Motivation of the Dissertation

The evolution of robotic systems has been rapid within the last few decades. Until recently, a robot was a machine which performed a repetitive task in a factory. Now, robotic systems can be found performing rescue operations, milking a cow, cleaning hospital floors, or even at home, performing services for people in their everyday lives. This new wave of robots is called field and service robots. The fundamental difference to industrial robots is that these robots are designed to be mobile and in many cases operate among humans as autonomous machines.

According to the International Federation of Robotics (IFR) [102] in the year 2006 there were 951 thousand industrial robots, 40 thousand service robots in professional use, and about 2.4 million service robots in domestic use in the whole world. The projection for the years 2007-2010 promises about 35 thousand new installations of professional service robot systems and about 3.6 million service robots sold for personal and private use.

The numbers of robots for domestic and private use have exploded during the past few years. The reason for this is the appearance of several companies providing low-cost robots such as vacuum-cleaning, lawn-mowing, and entertainment robots (including toy robots, hobby robots, and educational robots). These robots are simple, small in size, and designed to perform a single task, which makes them suitable for volume markets (cf. figure 1.1).

The robots targeted for professional use come in smaller numbers, but represent a much wider spectrum of applications, including defence, rescue, and security (23%), agricultural robots (mainly milking robots) (16%), cleaning robots (14%), underwater systems (14%), construction and demolition robots (10%), and medical robots (9%), followed by general robot platforms, logistics systems, inspection systems, and public relations robots. [102]

The figures show that robots are not yet widely used in professional markets. One clear reason for this is that most real-world applications are complex. The tasks

Figure 1.1: An entertainment robot and a vacuum cleaner robot operating in a home environment

require the handling of parameters and exceptions that simply cannot (yet) be composed into a fully automated system. This problem is widely acknowledged in the research community. An alternative approach is to use a combination of human intervention to replace the parts of the task that cannot be automated (or are not economically viable). A good example is the Sandvik AutoMine. It is an automated loading and hauling system for underground hard rock mining [144]. In the AutoMine system a loader (a heavy-duty vehicle with a bucket system) automatically navigates between the loading and discharge points. The loading phase, however, is conducted by an operator. This solution has been adopted mainly because the loading is the most dynamic aspect of the work cycle and no proper solution has been found to automate it. However, the solution makes it possible to use unmanned vehicles, moving the driver into the comfort of the control room. Additionally, the operator is controlling the machine only for a fraction of a work cycle, which makes it possible for him/her to control several machines at the same time.

More generally, there is a field of study called Human-Robot Interaction (HRI), which, in the words of Goodrich and Schultz [56] "is dedicated to understanding, designing, and evaluating robotic systems for use by or with humans". The idea behind HRI-based systems is that through interaction human and robot capabilities can complement each other, leading to more efficient and simpler systems, which can perform more complex tasks. HRI can further be divided into: 1) remote interaction and 2) proximal interaction. Remote interaction has a longer history in the area of teleoperation/telerobotics, but lately it has also adopted new aspects (such as the Sliding Autonomy concept [131]).

Proximal interaction requires humans and robots to be colocated (operating in the same space). The interaction can still take many forms (e.g. local teleoperation or natural communication between a human and a robot), but the recent trend has

Figure 1.2: Human-robot collaboration demonstrating the concept of working with the machines.

been to see robots and humans working together as team-mates. Conceptually, the idea is very different from an autonomy-centred approach, as humans are no longer just operating the machines, but *working with the machines*. Figure 1.2 illustrates the concept. The left-hand sub-figure shows a human guiding a robot to its local worksite by simply leading it. During the leading period, the robot is only required to control its actions in response to the human, who acts as the eyes and ears of the robot. What this means is that the robot might be able to perform e.g. a certain gardening task automatically, but it does not have to be able to move autonomously in arbitrary environments. The right-hand sub-figure shows an automated tractor bringing sand to a location specified by a human. Again it is possible to automate the task of getting sand from a specific pile, but to automate the whole task which requires this sub-task is difficult or impossible (for example, "fill this ditch", "build a road" etc.).

This introduces the concept of *Teamwork-centered autonomy*. Bradshaw et.al. declare solemnly in [13]:

> "A teamwork-centered autonomy approach takes as a beginning premise that people are working in parallel alongside one or more autonomous systems, and hence adopts the stance that the processes of understanding, problem solving, and task execution are necessarily incremental, subject to negotiation, and forever tentative. Thus, a successful approach to teamwork-centered autonomy will require that autonomous systems be designed to facilitate the kind give-and-take and richness of interaction that characterize natural and effective teamwork among groups of people."

In a sense the give-and-take approach in human-robot collaboration is even more natural than in human-human collaboration. An automated machine can be very productive, and can perform some tasks faster and more accurately than humans can. What a machine lacks is human reasoning, such as the ability to adapt to new situations, react to exceptions, and reason situations or plans as a whole. In the above examples (Figure 1.2) the human has the overall plan ("today we will do some gardening here" or "we need to even out this part of the yard") and he shares it with

the robot through interaction. This is the heart of teamwork-centred automation. Just as in human-human collaboration, not all the members have to have the same abilities. Different team members have different roles, which contribute to the team's overall performance, and the overall goal can be achieved through interaction.

As a concept, Teamwork-centred automation is ideal. However, from the technical point of view there are major challenges that need to be solved. Unlike in human-human teamwork, a human-robot team cannot be realised by simply saying "You two team up and perform this task". Humans possess natural mechanisms for interaction, such as the ability to communicate with each other and the ability to understand the environment and objects in it in a similar way and, up to some level, they understand what skills other team members have. This ability makes possible something which is called common ground. Common ground is defined as "knowledge that the participants have in common, and they know that they have it in common" [107] and is considered to be the cornerstone of successful teamwork.

There is no natural common ground in human-robot collaboration. Humans and robots have very little in common by nature and therefore this becomes an engineering problem. The common ground for human-robot teamwork is a synthetic product, with a set of requirements, design goals, and specific implementations. It also means that there is no single solution to the problem. Different requirements and assumptions will lead to different approaches, none of which are necessarily more correct or better than the others. There are, however, fundamental criteria that always need to be taken into account in the design. For example, any interaction requires some sort of communication, that is, the exchange of information between parties. The information should be understood by both parties (it is not difficult to make a robot ask a question; the problem is how to make the robot understand the answer). The tasks of a human-robot team are most probably bound to the physical world. This means that the task-related objects and the workspace in general should be understood by all parties. Furthermore, in a spatially distributed team the information is tightly coupled into a frame of reference. Different team members possess their own coordinate systems and for the team the information is useless unless others can transform the information into their frame of reference.

1.2 Case Study: PeLoTe Project

Much of the research reported in this thesis was carried out within a research project called PeLoTe. PeLoTe is an abbreviation for "Building Presence through Localisation for Hybrid Telematic Systems" and it was a European Union-funded project, IST-2001-38873. The project was active between 2002 and 2005. The objective of the PeLoTe project was to investigate, formalise, and develop methods for the coordination of telematic systems involving humans and autonomous or teleoperated mobile robots. PeLoTe was a general system study, but the research was grounded with a proof-of-concept search and rescue scenario. A demonstrator called the PeLoTe system was designed, implemented, and demonstrated in a simulated fire-fighting scenario (see Figure 1.3).

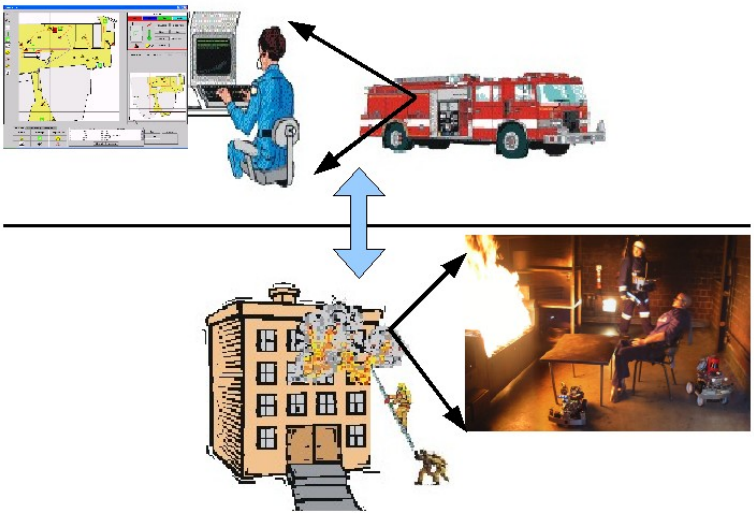


Figure 1.3: PeLoTe case study. A team of humans and robots performing a search and rescue task with an operator's help.

The PeLoTe system is a telematic multi-entity system; that is, it is a remotely controlled system, which has more than one actor. The system consists of an operator (more precisely, a coordinator) and telematic entities (humans and robots), working together in a remote location.

The simulated mission started with a fire alarm to a fire station. On the way to the scene the mission leader downloads the so-called Standard Rescue Map (SRM), which is a layered map containing the a priori information on the building in question (cf. [94]). The mission leader used the a priori information to generate an initial rescue plan. A *cooperative planning system* (cf. [82]) was used as a tool to generate initial trajectories for different entities.

On arrival at the scene, the exploration team was launched to inspect the building according to the initial plan. The goals of the task were to recover the state of the environment, put out all the fires, rescue all the victims, and get the whole team safely out of the building.

In the PeLoTe project only a simple solution to the given problem was realised. The level of autonomy in the system was moderately low. Nevertheless, the system demonstrated a human-robot team performing a search and rescue task. The simplicity arises from the integration of humans into the system. Both the operator and the human rescuer have a significant role in the system. The high-level decisions in the system are left to the humans. The system in turn is built to support this.

The common ground in the system is built on the a priori knowledge. First, all information is bound to a common model, which uses a single frame of reference to represent the data. The a priori map of the building is the basis of the common model. Second, the task is defined as the exploration of a partially unknown environment, which sets certain design criteria (for localisation and planning). The common model consists of a limited set of objects (or zones) that everyone in the

system should know. For humans this requires training and for robots (or the global planning module) the objects have preprogrammed data association models.

The common model can be maintained only if all information is reported with respect to the given frame of reference. This is possible only if all the entities are able to orient themselves with respect to that frame of reference. One way to achieve this is to localise all the entities, including the human ones. This provides the major motivation for this thesis. The human entity who was working on the operational level was also part of the system. From the system point of view the human cannot contribute much without additional interfacing. The human was added to the system by developing a Personal Assistance System (PAS). PAS consists of a personal localisation system called PeNa and a user interface that allows the human to report his/her findings and to follow the given task. PeNa is a portable sensor system that derives the position of the human entity within the shared model. The location is required to help the human entity to orient him/herself, but also to help the operator to coordinate the actions of the human entity.

During the mission, the common model is constantly updated by both the humans and robots. A robot could add sensor readings to the model directly, or it could send an event to the operator, who would add an object to the map on the basis on the sensor readings that had been sent. Additionally, the human entity could add findings him/herself on the basis of the current position or by informing the operator using voice communication, who in turn adds the object on the basis of the position provided by PeNa. Human-augmented mapping incorporates the power of human reasoning into the system. While robots use laser scanners and other sensors for accurate numerical mapping, humans use their ability to put pieces together. For example, the human entity can report to the operator that "the corridor in front of me has collapsed" and the operator marks the corridor as a restricted area on the map.

During the mission, the common model incorporates the a priori information and information integrated during the mission, as well as the states and tasks of the entities. In a sense, the common model can be understood as a model of shared situation awareness for the whole team (including the operator). The model holds everything that is relevant and that it is necessary to know (or that is known) for the team to perform its mission.

1.3 Problem formulation

The main problem to be studied in this thesis is how to incorporate humans into a human-robot team performing a search and rescue-type task as described by the PeLoTe project. The exchange of information in the system is based on a common model, which requires the information to be provided in a single frame of reference. Therefore the beginning premise is that all team members must be localised with respect to a given a priori map. The location information is used to adapt both the human team member and the operator into the global frame of reference,

which makes human-augmented mapping possible, as well as providing navigation assistance for the human.

Naturally, humans navigate relative to the environment without knowledge of their numerical coordinates, which means that they cannot tell the system their positions. Therefore the human needs to be equipped with an additional navigation system, called PeNa. The design of the system is motivated by the PeLoTe project and therefore also by the simulated search and rescue scenario. The following requirements and assumptions were considered in the design phase:

- PeNa needs to localize a human in indoor conditions;
- there is an a priori map of the building;
- the initial position is known approximately;
- the localisation is based on self-contained sensors;
- no preliminary installations or calibrations can be made on-site;
- the localisation is performed in a single plane, that is, in 2D.

This thesis presents methods, in combination with different sensors, which lead to a solution that satisfies the given requirements and enables a human to be incorporated into a human-robot team performing a difficult search and rescue scenario.

1.4 Main Contribution of the Dissertation

The main contribution of this thesis is the new methods developed for personal navigation. The navigation methods are divided into personal dead reckoning methods and map-based methods.

The dead-reckoning used is based on the following methods:

- use of a foot-mounted step length-measuring device based on direct measurement of the distance between the ankles;
- use of step length estimation based on 2D accelerometers mounted on feet;
- combining step length estimation with estimated heading to obtain a 2D trajectory.

The dead reckoning result is further refined by using a laser range finder. The applicability of using the laser for personal navigation is studied and the following two alternative methods for computing refined pose are presented:

- pose correlation-based scan matching method;

- combined angle histogram matching and 2D position correlation method.

Dead reckoning can provide accurate short-term information about the movements of a person. The long-term positioning is obtained through reference measurements. In this case an a priori map is used as a reference. The map-based localisation is based on Monte Carlo Localisation (MCL), which is a particle filter-based method. Two basic approaches to map-based localisation are presented:

- topological MCL, which only matches the motion to the map on the basis of dead reckoning
- laser-based MCL, which matches the laser measurements to the map.

Additionally, these two methods can be combined. The map for the methods is expected to be partially correct. The methods expect most of the structures to be on the map, but none of the temporary features need to be on the map (furniture, closets etc).

Finally, the personal navigation system that was developed is integrated into a Personal Assistance System (PAS), which is used to incorporate a human into the PeLoTe system. The PeLoTe system demonstrates a human-robot team performing a search and rescue task. The results of the experiments performed with the PeLoTe system are reported and analysed, with a special emphasis on assessing the influence of the personal navigation system on the whole system.

While most of the individual mathematical methods used in this thesis have been published before, this thesis contributes to the research community by:

- formulating a topological Monte Carlo Localization method;
- providing a unique personal navigation system that can localise a human indoors with self-contained sensors;
- using the personal navigation system as a tool to interface a human into a human-robot system.

All items are considered novel. As far as the author is aware, there are no infrastructure-free personal navigation systems that can provide a bounded error position estimate in indoor conditions. Furthermore, this thesis provides a unique viewpoint for human-robot team interfacing. The use of the location of a human provides valuable information to the operator coordinating the team. It also enables spatially bound information to be shared in a single frame of reference.

Finally, this thesis formulates a method for map-based localisation, which only uses a map and dead reckoning as input and outputs a bounded error pose estimate. Similar work has been presented earlier (e.g. by Kane [106] and by Sang et al. [103]), but as far as author is aware there is no probabilistic formulation, with experimental results, presented earlier, which would be able to provide a pose estimate based on erroneous dead reckoning and a non-perfect map.

1.5 Thesis Outline

The contents of the thesis are structured as follows:

- Chapter 1: a brief introduction to the subject and the research objectives of the thesis.
- Chapter 2: a brief introduction to central concepts and the state of the art regarding human-robot teams.
- Chapter 3: review and analysis of sensors and methods applicable for personal navigation and review of the current state of the art regarding personal navigation systems. This chapter presents a broad overview of the field of personal navigation and analyses the methods used in different solutions by other researchers.
- Chapter 4: novel methods for sensor-based personal navigation. This chapter presents the new methods and approaches developed for sensor-based personal navigation. Additionally, the test system is introduced in detail.
- Chapter 5: tests and results. This chapter presents the results of different localisation methods, individually and integrated. Finally, the tests performed with the complete PeLoTe system are presented and analysed.
- Chapter 6: the main results are summarized.

1.6 Author's contribution within research groups

The major part of this work was performed in the Building Presence through Localisation for Hybrid Telematic Systems project (PeLoTe, IST-2001-38873). The personal navigation system was developed by the team led by the author. The whole system and most of the algorithms were designed by the author. The other contributions were from Seppo Heikkilä, who implemented the final version of the scan matching [65] and Mikko Elomaa, who implemented the Stride length measurement units [43]. Additionally, Frauke Driewer was responsible for the development of the GUI for the Personal Assistance System [35, 34]. The method for computing the reference trajectory for dead reckoning was made by Janne Paanajärvi.

The PeLoTe system was a joint effort by researchers from five different institutes. The final experiment was therefore a joint effort by everybody. The results were documented in a European Union project deliverable, which was agreed to be freely usable by the research group. The analysis and conclusions based on the experiment results, presented in this thesis, are made by the author.

1.7 Declaration of previous work

Part of the work reported in this thesis has been published in several publications. The most relevant publications to which the author has contributed are as follows:

- the development phases of the personal navigation system have been published in [118, 119, 121, 115, 116];
- the overall system has been published in [83, 37, 120];
- additionally, the concept of the common model/presence has been reported in [135, 134]

Additionally, the PeLoTe experiment has been published in [35, 125, 109]. Moreover, in her PhD. thesis Driewer [36] studies the user interface design for human-robot teams.

Chapter 2

Introduction to Human-Robot Teams

In this thesis the human-robot team is understood as a mixed team of humans and robots working together in a shared environment to achieve a common goal. The team is coordinated by a remotely situated human operator. The human-robot team that performs the actual work is called a remote team. A human team member is referred to as a human entity (HE) and a robot team member is called a robot entity (RE). The setup offers two viewpoints for the thesis: 1) the remote coordination of mixed human-robot teams and 2) cooperation in mixed human-robot teams. The technology which makes the remote coordination of the mixed teams possible is studied, within the limits of this thesis. The cooperation of mixed human-robot teams is a much wider research area, and therefore is mostly left for future work.

2.1 Human team work

Human teamworking has been studied a lot and, within this thesis, only a few key results significant for this work are addressed. It has been noted that spatially distributed teams have more difficulties (or less productivity) than closely situated ones (c.f. [1, 107, 12, 27]). This is especially true for teams that are completely separated from each other, but also for teams that share the same office. For example, in his studies Allen [1] shows that the probability of communication declines as a function of distance between two team members. Olson and Olson [107] analyse work teams that are colocated and teams that are remotely located. They emphasise that common ground between effective work teams is important. Common ground is defined as "knowledge that the participants have in common, and they know that they have it in common" [107].

Common ground is an intuitive concept; the more you have in common with someone, the easier the collaboration will be. For example, one explains the same concept to a co-worker quite differently than to a stranger. In a conversation between coworkers one can assume that the other person knows the same terms as you, and probably has the same education and cultural background, while in a conversation with a stranger, one has to create the common ground simultaneously by explaining

the background more thoroughly. In the creation of the common ground co-presence has a significant meaning. During conversation, the visual cues of the listener can reveal if he/she understands what is being explained and give an opportunity for the explainer to revise the explanation. The co-presence of communication partners is also significant because it makes it possible to use spatial references (this, that); in other words, the participants share the environment and the objects in it, which simplifies the discussion.

Jones and Hinds [70] observed the field training of Special Weapons And Tactics (SWAT) teams. SWAT teams are special units that are trained to perform high-risk operations, including hostage rescue. A SWAT team is an example of centralised decision-making in a distributed information network. The SWAT team consists of field teams (an assault team, hostage negotiators, and sniper teams) and commanding officers (an incident commander, tactical commander, hostage negotiator leader, and logistics supervisor) [70]. Different teams are spatially separated, e.g. the assault team penetrates into a building or some other area, while the sniper team is outside the incident area but within visual range. The command centre is at some distant place away from the incident area.

In their findings Jones and Hinds emphasise the importance of common ground. The common ground is partially created by training (terminology, communication procedures, and standard sets of procedures). The common ground is further grounded by a face-to-face meeting, which is called a Situation Report. During this meeting the situation is described (description of the enemy, the environment etc) and roles for different members are given. Additionally, street maps and building floor plans are given out. Thus, when the team moves to the incident site, everyone knows their own task, as well as having information about the movements of the others, and they have an initial frame of reference (in the form of maps).

During the mission the teams were reported as having lost sight of each other, which caused some confusion if the teams tried to coordinate with each other. In fact, Jones and Hinds observed that the role of the tactical commander in maintaining the common ground during the mission was significant. The conversations mostly took place between the teams and the commander (not between the teams). The commander had the most complete picture of the situation, which he updated for the team members if necessary. An important finding was that communication between the tactical commander and team members took place in the team member's frame of reference (e.g. "Do you see a stack of boxes to your left?"). The tactical commander maintained the situational picture in the global frame of reference and when referring to some unit the commander took the viewpoint of the unit.

Another related and widely researched concept is *situation awareness* (SA). Informally, situation awareness is "to know what is going on around you". Endsley has presented a widely accepted formal definition (c.f. [44, 45, 71]):

> "The perception of the elements in the environment within a volume of time and space, the comprehension of their meaning and the projection of their status in the near future."

Endsley defines SA by using three levels. First, one needs a perception of the relevant situational cues, and second, one needs to understand the relevance of these cues to the situation. Finally, if the two former levels are fulfilled, he states that "the ability to forecast the future situation events and dynamics marks operators who have the highest level of understanding of the situation".[45]

SA has its roots in aviation, but has spread to practically all fields that require the control of dynamic processes [40]. SA has also been extended to teamwork. A team is a sum of individuals working to achieve a common goal. Each team member has his/her own task and therefore creates his/her own situation awareness. Endsley defines situation awareness for a team as "the degree to which every team member possesses the SA required for his or her responsibilities" [44, 71]. The SA of individuals may overlap, but the SA of the team is more than that of the individuals in it. The overlap is often required because of the interdependency of the team. In a functioning team each member needs to share a common understanding of what is happening regarding those aspects that are common – shared SA. Formally, Endsley defines shared situation awareness as "the degree to which team members possess the same SA on shared SA requirements".

Situation awareness and common ground notations are closely coupled. In this thesis a difference is drawn according to the dynamics of the situation. Common ground is something that is known a priori, while situation awareness is an understanding of what is happening at a particular time instant in some system.

2.2 Human-robot teams

The area of human-robot collaborative teams is relatively new. According to [58], one of the earliest pieces of work done on human-robot systems working in collaboration was introduced by Inagaki et al. [67] in 1995. The work introduces a system that is based on intention inference. In the experiment a human is remotely operating one robot and, on the basis of the predicted intention of the operator, another autonomous robot adjusts its own rules in a simple cooperative task of moving desks from one formation to another. This simple test demonstrated some of the key issues in human-robot collaboration that are still valid. The intention inference is one of the keys to building the common ground between the human and the robot. The work also demonstrated that through a well-regulated task it is possible to predict the intentions of the other cooperating party.

Currently, the research on collaborative robotics is mainly driven by military applications (cf. [76, 6]), future visions on space applications (cf. [50, 49, 51, 131, 133]), and search and rescue (cf. [18, 100, 17]). Interest in applications aimed at consumer markets, such as home robotics [86] and assistive robotics [145], is also growing.

Little work had been done on the coordination of mixed human-robot teams until recently. NASA's peer-to-peer human-robot interaction project [50] has been developing a framework for collaboration between humans and robots called the Human-Robot Interaction Operating System (HRI/OS). They demonstrated the system in

a simulated construction task, where the work was performed by two humans and two robots working as a team and an operator who provided support remotely.

Bradshaw et al. have demonstrated a human-robot team for searching for an intruder from a Navy pier [13]. The search team was composed of one human and five robots and was coordinated by a supervisor. The search teams were further divided into sub-teams that had a leader and members that had specific tasks (or a hierarchy). The sub-team leaders could be robots or humans.

A similar experiment, but on a smaller scale, was performed by Kennedy et al. [76]. The task was to follow a moving target (a walking human) with a human-robot pair, so that they would remain invisible to the target but would maintain visual contact with the target. In the experiment the robot was acting autonomously, but could be commanded with simple speech or gesture commands ("Come here", "Stop"...).

In all three projects mentioned above([51, 13, 76]) the robot understood speech and at least in [51] was responding with speech. On the other hand, the robots most probably did not understand natural language; instead, the operator had been trained to use these robots with specific commands. In other words, the human adapts him-/herself to the robot's world. This is most often achieved by offering a user interface and instructions (training) on how to use it. The design challenge is to make the interfacing natural for a human so that the interface is easy and effortless to use.

In section 2.1 it was mentioned that spatial referencing helps the grounding in human-human collaboration. Spatial referencing means that people can e.g. point to an object and speak about it, e.g. as "that object" or "the object is over there", etc. The ability to do the same in human-robot collaboration is just as important. The fundamental difference is that humans working together share the ability to comprehend the environment (including objects and abstractions within it) in a similar way. For example, humans naturally label objects and places ("red chair", "living room", "glass of water"), which provides an efficient means to refer to them. This provides a huge common ground for human-human interaction.

Another significant ability of humans is perspective-taking. In natural discussion, a human may orient him/herself into several frames of reference without even thinking about it (such as "my left, your right"). Jones and Hinds [70] reported incidents from the SWAT training which showed that a highly trained professional can fuse multiple egocentric frames into one exocentric frame and switch fluently between the frames while communicating with the crew members. What was even more remarkable was that the tactical commander was able to do this without visual contact with the crew members.

To model such capabilities for a robot is a challenge that will not be met in the near future. However, in limited tasks some of the abilities can be modelled. Fong et al. [51] used a separate Interaction Manager for the coordination of communication between humans and robots. The interaction manager allows dialogue-based communication and supports a graphical user interface. The spatial referencing in dialogues is handled by a Spatial Reasoning Agent. The agent is based on the Stage

Simulator [55], which maintains the model of the environment in a single frame of reference based on the sensor data and object trackers. On the basis of the model, the Spatial Reasoning Agent is able to switch between egocentric, referent-centric, and exocentric frames [49]. As a result, a sentence such as "go to the left of box n" will result in spatial coordinates for the robot.

Bradshaw et.al. present a similar solution in [13]. They implement a shared global model of the environment (KAoS Spatial Reasoning Component). It allows the operator to create objects and label them (e.g. "this is a shed"). The labels can be further used in spatial referencing. Moreover, the component has a similar spatial reasoning capability to that reported in [51].

One basic need in human-robot collaboration is to have a shared representation of the environment. In the above projects the shared model (even though it might not have been directly visible for a human) was a global map, with all the object locations and labels. In [13] the operator had the ability to use his/her own cognition to add items to the representation based on sensor data coming from a robot.

Kaupp et al. [74] present a work in which a model-based representation is used to fuse the information coming from the robots and operators. The representation is a feature-based probabilistic model, with object positions and a visual model, which provides information for classification. The human incorporates the information by giving e.g. the estimated position or range of an object and a feature class (in this case there were only four options). As a result the operator was able to participate in the mapping process. The work demonstrated the possibility of a human-robot team complementing each other. A human operator is not accurate in e.g. measuring the range to an object. However, a human is superior to a robot in a classification problem.

In peer-to-peer interaction a human should be able to understand the actions of a robot. As an example, if a heavy robot approaches a human with deadly speed, the human might be interested to know whether the robot is aware of his/her presence or not. Similarly, as stated by Bradshaw et al. [13], it could mean that if a robot is sent to perform a task, a human (or just as well a system) can rely on the robot reporting after completing the task.

On the other hand, situation awareness can be understood as the ability to find out why the system is in some state. In particular, this ability is required when robots request help from humans. Human-robot teamwork introduces a new paradigm of automation. A robot may not only possess one level of automation. Instead it may be able to vary the level continuously. This concept is known as Sliding Autonomy [131]. The sliding autonomy concept allows a robot to ask for help from the operator, just as the operator can take control of the robot (called Mixed-Initiative Sliding Autonomy (MISA)). One of the important research areas is to be able to give the operator sufficient information about why the robot has asked for help. As an example, Sellner et al. [131] demonstrated an assembly task with multiple robots and a human supervisor. The system (the robot assembly team) could perform autonomously, but from time to time could request assistance from the operator. In their case the situation awareness was achieved by having an overhead

video which was played back. Similarly, in the work presented by Fong et al. [51] they implemented a module called the context manager that logged everything that happened in the system for later playback.

Situation awareness in the traditional sense is met in the coordination of the team by a supervisor. A supervisor needs to know "what is happening in the whole system". Jones et al. [70] showed that coordinating a remote team of humans is possible with only audio information. They also reported that, to be able to do that, the team leader combined several sources of information into an exocentric frame of reference (or whatever structure resembled this in the leader's mind). This seems to be the only solution when coordinating teams that are not physically working close to each other (colocated, but distant from each other). The global bird's eye view is widely used in this kind of application (cf. [13, 145, 15]) and in fact, even Fong et al. [51] did not use the map as an interface in collocated interaction, but it was used in background (Stage simulator).

Finally, what should the shared situation awareness in a human-robot team be? As in human teams, the shared situation awareness should be the relevant information that it is necessary to share in order to successfully accomplish the given task. For example, in an information-gathering task (e.g. [74]) the map is shared among all the team members. Similarly, in the exploration of an unknown environment, the map and the future goals of others should be shared in order for one member to effectively plan future activity.

Chapter 3

State of the Art Regarding Personal Navigation

3.1 Terminology and methodology

Personal navigation systems derive and provide information about the location of humans. In the literature the terms Pedestrian Navigation System [69] or Personal Positioning System [2] are also used. The term Pedestrian tracking is also used [141, 54], but it is more commonly used in the context of tracking humans from or by means of a vehicle (cf. [123]). Personal navigation systems are based on portable devices and sensors which make the localisation possible. Personal navigation systems are further categorised by different localisation approaches. An infrastructure-based system takes advantage of existing or specially designed infrastructure in the localisation process (such as GSM, WLAN, or GPS), while infrastructure-free methods are based on information derived from devices that are carried by the user.

Personal Dead Reckoning (PDR) is a special type of system which provides a relative position estimate by tracking the differential movement of humans over time and provides an integrated position as a result (for example by estimating step length and the number of steps taken).

3.1.1 Frame of reference, maps, and localisation

Localisation is a process of determining one's position with respect to some frame of reference. The frame of reference is defined by a coordinate system and its origin. For example, Global Positioning System (GPS) uses an Earth-fixed spherical coordinate system, which specifies a position in terms of longitude, latitude, and distance from the centre of the Earth. In general, the frame of reference can be fixed to any local (or global) coordinate system and a position can be addressed with respect to it.

A map is defined as a description of a physical environment for a specific application [52]. A map represents the environment by some projection and simplification. For

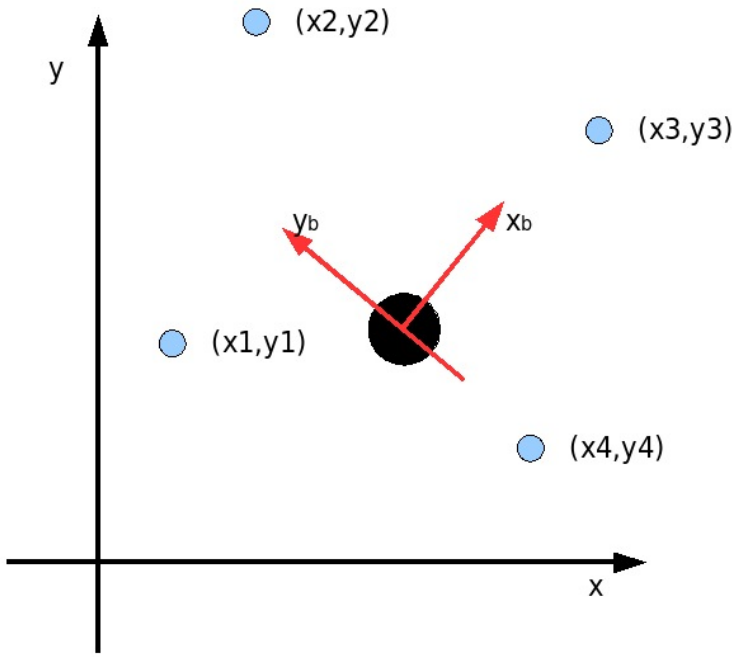


Figure 3.1: An example of a feature map in the Cartesian coordinate system

localisation the representation of the map should be such that the observations can be associated with the map features and by doing so the location of the observer can be derived. For example, in Figure 3.1 a two-dimensional feature map is represented in a Cartesian coordinate system. The map features are given with respect to the origin of the frame of reference as:

$$m = \left\{ \begin{array}{c} (x_1, y_1) \\ (x_2, y_2) \\ (x_3, y_3) \\ (x_4, y_4) \end{array} \right\}. \tag{3.1}$$

To localize the observer (black dot in Figure 3.1) with respect to the map, the features must be observed by some sensor. A sensor measurement is usually transformed to a range or bearing or combined measurements to map features. The measurement must also be associated with the map features; that is, each measurement (range or bearing) should be assigned to a specific feature. This is called data association. Because the sensor is carried by the observer, the measurement is performed in the body's frame of reference ((x_b, y_b) in Figure 3.1). To define a position uniquely in 2D, in the case of Figure 3.1, one needs: 1) three range measurements (trilateration [11]); 2) three bearing measurements (triangulation [11]), or 3) two combined angle and bearing measurements with reference to known features.

In Figure 3.1 the map features are so-called point features. The point in a physical environment can be any object which can be detected as a point, for example a pillar, a tree, or a corner. In general feature maps represent the environment by

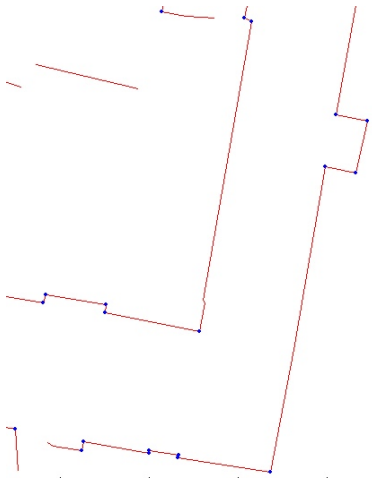


Figure 3.2: Feature Map, with lines and corners

means of extracted features. These features can be any suitable landmarks for the selected sensor and environment (e.g. trees, walls, corridors, corners, or boxes). The landmarks (features) are presented with their parameters in the global coordinate system. Figure 3.2 shows an example of a feature map which has two kinds of extracted features: corners (point objects) and lines are given.

Another type of map representation is an occupancy grid map. An occupancy grid [96, 42] or evidence grid [126] represents the environment by dividing it into cells. Each cell has a value that indicates if the cell is occupied by an obstacle. Usually, an occupancy grid is used in a probabilistic manner, which means that each cell has a state that states the probability of that cell being occupied. The estimation of each cell is usually decomposed to static estimation for each cell (that is, the cell probability depends on other cell values) using binary Bayesian filtering ([138] pp. 94, 284-293).

Bayesian filtering allows the use of a sensor model to represent the measurement uncertainty. This makes the occupancy grid suitable for mapping with inaccurate sensors, such as ultrasonic sensors.

In Figure 3.3 an example of an occupancy grid and a sensor model for a range sensor is given. The black area is free, the white one is occupied, and the grey one unknown. The right-hand image shows the "sensor" model for an accurate range sensor. The area between the robot and the obstacle, reported by the sensor, is obviously free space.

In general, mapping is a process of transforming the sensor observation into map features (or probabilities in the case of an occupancy grid) using appropriate sensor models. Mapping requires the position from which the observation was taken to be known. Simultaneous Localisation And Mapping (SLAM) is a problem of estimating both the location and the map simultaneously.

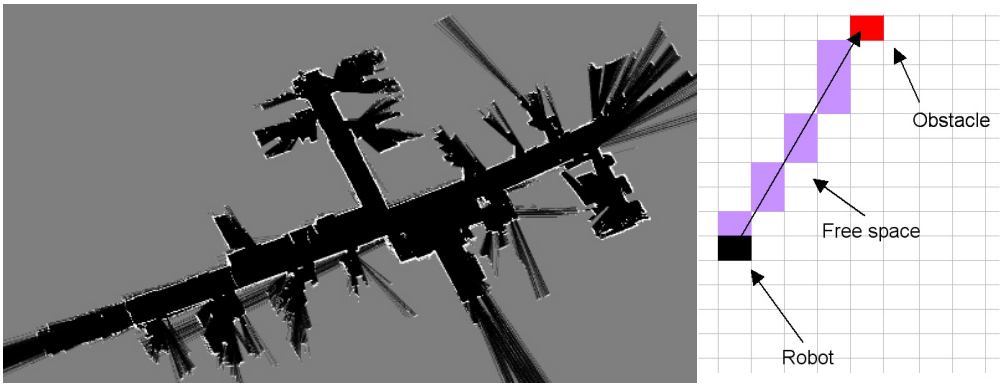


Figure 3.3: Occupancy grid and "sensor model" examples. Left: occupancy grid of an office corridor. The white cells are occupied cells (walls and obstacles) and the black cells are free space. The grey area is unmapped/unknown. Right: a sensor measurement is added to the map by updating both the free space (which the range "beam" has passed through) and the obstacle as occupied (point of reflection).

3.1.2 Continuous localisation

Localisation can be divided into two different approaches: global and continuous localisation. Global localisation can define the location without any previous knowledge of the position. For example, GPS can pinpoint you anywhere in the world in a matter of minutes from starting up the receiver. Continuous localisation is a problem of tracking position over time. This is a state estimation process. At a given time t, the robot is in some specific pose x_t (position and orientation; the notation p_t is also used later) with respect to some frame of reference. In this thesis a 2D pose is considered, that is:

$$x_t = \begin{pmatrix} x(t) \\ y(t) \\ \varphi(t) \end{pmatrix}, \tag{3.2}$$

where $(x(t), y(t))$ is a position of the localized entity in Cartesian coordinates and $\varphi(t)$ is the orientation of the body frame with respect to the frame of reference.

The state x_t is the result of a series of executed controls:

$$u_0, u_1, ...u_t = u_{0:t} \tag{3.3}$$

which has resulted in a set of past states:

$$x_0, x_1, ..., x_t = x_{0:t}, \tag{3.4}$$

in each state an observation is made:

$$z_1, z_2, ..., z_t = z_{1:t}. \tag{3.5}$$

A convenient estimator for continuous localisation is a recursive estimator, which estimates the next state on the basis of information about the previous state, controls, and the most recent measurement. In Equation 3.6 a prototype of a "general" state estimator for localisation in recursive form is shown:

$$\begin{cases} \hat{x}_t^- = f(\hat{x}_{t-1}, u_t) \\ \hat{x}_t = g(\hat{x}_t^-, z_t, m) \end{cases} , \tag{3.6}$$

where $\hat{x}_t$ is used instead of x_t to differentiate the estimate from the true state. The first equation corresponds to a *motion model*. A motion model is the input-output model for *predicting* the movement between measurements on the basis of the given controls. The second equation is an *observation model*. The observation model updates the prediction made with the motion model by using the most recent measurement and map m.

Continuous localisation can lead to a dead reckoning or absolute position estimate. The principal difference is that dead reckoning uses only the motion model ($f(\hat{x}_{t-1}, u_t)$ in Equation 3.6) for updating the state. Absolute positioning provides a position estimate with respect to a map by using $g(\hat{x}_t^-, z_t, m)$.

3.1.2.1 Dead Reckoning

Dead reckoning comes from an old term used in navigation and means the process of determining one's position from a compass direction and the estimated distance travelled (speed and time), but in general dead reckoning in robotics refers to the calculation of position from motion sensor feedback. The terms odometry and inertial navigation are used in a similar sense, but the difference lies in how the position is derived. Odometry refers to a process in which the robot's position is determined from wheel encoders (or similar sensors) and a kinematic model of the robot or vehicle. Inertial navigation refers to the usage of inertial sensors. Inertial sensors, such as a gyroscope, accelerometer, and inclinometers, give information about the inner state of a body. Dead reckoning gives an estimate of relative movement from the initial position. Thus it does not give an absolute position with respect to the map, but it shows how far and in which direction the entity has travelled from the starting point.

Basically, all three terms lead to the same solution. Given an initial state x_{t-1} the task is to estimate the movement at the interval $t-1 \rightarrow t$ and derive the next state. Basically, no matter what is the approach called (odometry, dead reckoning, or inertial navigation), one can compute a differential movement $(dx_t, dy_t, d\varphi_t)$, which is the differential movement from state x_{t-1} at interval $t-1 \rightarrow t$. This is illustrated in Figure 3.4a. The differential movement is calculated with respect to the body frame defined by the state x_{t-1}. The same movement is illustrated with respect to the world/navigation frame of reference in Figure 3.4b. The differential movement $(dx_t, dy_t, d\varphi_t)$ can be expressed in polar coordinates as:

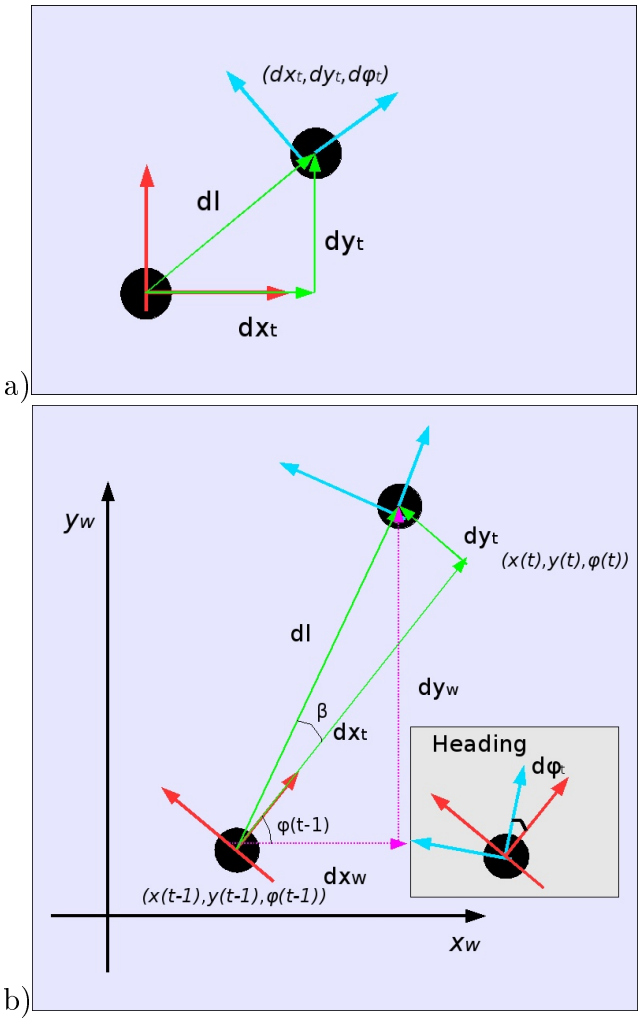


Figure 3.4: Dead reckoning principle. a) Movement in body frame of reference b) movement in global frame of reference.

$$\begin{cases} dl = \sqrt{dx_t^2 + dy_t^2} \\ \beta = atan2(dy_t, dx_t) \end{cases}, \tag{3.7}$$

where atan2 is an algorithm to compute the principal value arc tangent of y/x for given y and x in range of $(-\pi, \pi]$. The same movement in polar coordinates with respect to the world coordinate system can be expressed by the pair (dl, α), where α is defined as:

$$\alpha = \varphi(t-1) + \beta \tag{3.8}$$

The differential movement in Cartesian coordinates with respect to the world frame of reference is then:

$$\begin{cases} dx_w = dl\ cos(\alpha) \\ dy_w = dl\ sin(\alpha) \end{cases} \tag{3.9}$$

The change in orientation $(d\varphi_t)$ remains the same between the transformations and thus the new pose estimate can be given as:

$$\hat{x}_t = \hat{x}_{t-1} + \begin{pmatrix} dx_w \\ dy_w \\ d\varphi_t \end{pmatrix}. \tag{3.10}$$

Equation 3.10 is often used to replace the motion model in Equation 3.6. In this case the differential movement is used as the control signal:

$$u_t = \begin{pmatrix} dx_t \\ dy_t \\ d\varphi_t \end{pmatrix}, \tag{3.11}$$

where the differential movement is calculated with respect to the pose at time $t-1$, as shown in Figure 3.4a. This differential movement is always affected by drift, which means that the measured (or calculated) movement is a sum of the actual movement and measurement error:

$$\begin{pmatrix} dx_t \\ dy_t \\ d\varphi_t \end{pmatrix} = \begin{pmatrix} dx_t^r + e_{x_t} \\ dy_t^r + e_{y_t} \\ d\varphi_t^r + e_{\varphi_t} \end{pmatrix}, \tag{3.12}$$

where the superscript r is used to denote the real displacement from the previous pose and e to denote the measurement error (also called noise). The dead reckoning error is driven through Equations 3.7-3.10. Equation 3.10 is integrative, which means that every time the update is performed, the error is also integrated into

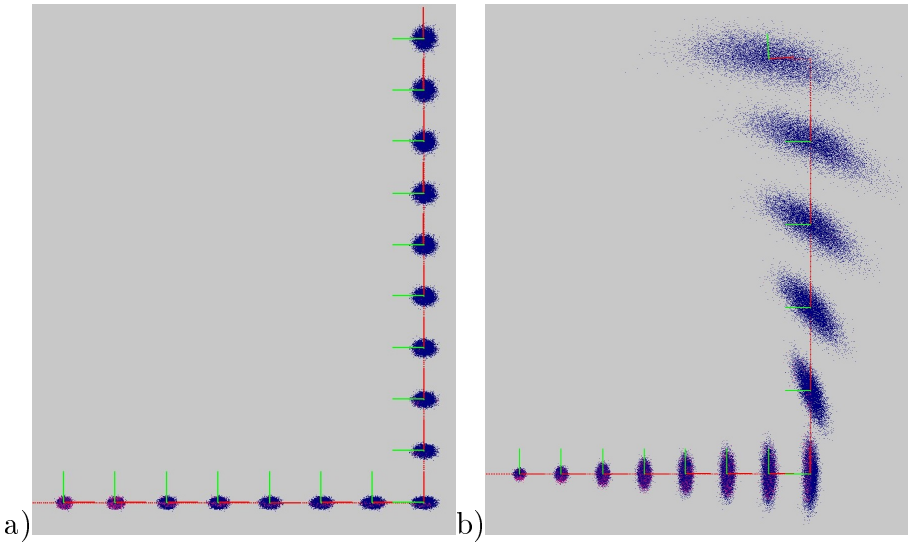


Figure 3.5: Example of noise evolution in dead reckoning. a) The noise is generated mostly in the direction of movement; b) the noise is generated in the direction of movement and in the heading measurement.

the result. Additionally, the resulting error in pose is non-linear, because of the non-linear transformations made.

Figure 3.5 illustrates the evolution of pose uncertainty in simulated movement. In the left-hand image the noise is mostly generated in the direction of movement, while in the right-hand image the noise is also induced in the heading measurement. The blue dots represent the possible poses at a given time instant. The estimation starts from the more concentrated clouds and as the simulated body moves forward, the uncertainty grows, which is visible as large clouds in the image. The shape of the clouds also depends on the shape of the trajectory.

This means that dead reckoning can only be used for short-term localisation methods. No matter how accurate the dead reckoning estimate is, the error will eventually grow beyond any given boundary. The dead reckoning error is typically expressed as percentages of distance travelled. For example, in the case of a simple robot with wheel encoders, the typical error in a dead reckoning estimate is around 1% of the distance travelled. However, the heading error depends mainly on the kinematics of the robot. The heading error largely affects the shape of the estimate (which is visible in Figure 3.5b) and therefore makes the estimation of the true dead reckoning error difficult. A relatively common practice is to measure the dead reckoning error as the displacement error of a round trip (travelling through some loop and ending up in the starting position). The problem with this type of measurement is that the position displacement does not reveal the true error, because the heading and displacement errors are coupled. Nevertheless, it gives an idea of the magnitude of the dead reckoning error and it is therefore used as one measure of accuracy in this thesis.

3.1.2.2 Probabilistic map-based localisation

As illustrated in the previous section, the dead reckoning error grows without limits. Additionally, dead reckoning only estimates the movement with respect to the initial frame of reference. In most of the applications the pose must be calculated with respect to the given frame of reference (often defined by the map) and the accuracy of the pose estimate needs to be within the given limits. To be able to do this, one needs information about the absolute pose with respect to the given frame of reference. This information is obtained by measuring the actual state of the environment with a sensor(s). The map-based localisation problem is to find the pose that best fits the measurement on the given map. Continuous map-based localisation tracks the localised entity continuously, so that the fitting is done locally in the surroundings of the estimated pose. Basically, the accuracy of the dead reckoning defines how large the search area should be or how long the intervals without any updates can be.

The biggest challenge for localisation is uncertainty. The models are never perfect, which causes errors in predictions; sensor measurements are noisy (some more than others) and the environment model is an approximation. Additionally, in reality the environments are rarely truly static (for example, moving people, changing furniture, new radio beacons etc.). Therefore, all successful approaches to localisation are basically probabilistic. This means that instead of the state being estimated directly, the probability distribution of the state is estimated [138]:

$$p(x_t \mid z_{1:t}, u_{1:t}, m). \tag{3.13}$$

Equation 3.13 gives a conditional posterior probability distribution of state x_t, which at time t holds all the information about the earlier controls and measurements. This probability distribution is also called as belief [138]. The update of the distribution given by Equation 3.13 is called Bayes filtering. The applications of Bayes filters applied into localisation are called *Markov localisation* [138]. Markov localisation dates back to the mid-'90s and is currently widely used in robotics (cf [53, 99, 19, 16]). The updating of Equation 3.13 can be formulated as a recursive procedure, which inputs the previous posterior, current control and measurement (derivation can be found e.g. [138] pages 31-33):

$$\begin{cases} p(x_t \mid z_{1:t-1}, u_{1:t}, m) = \int p(x_t \mid x_{t-1}, u_t, m) p(x_{t-1} \mid z_{1:t-1}, u_{1:t-1}) dx_{t-1} \\ p(x_t \mid z_{1:t}, u_{1:t}, m) = \eta p(z_t \mid x_t, m) p(x_t \mid z_{1:t-1}, u_{1:t}) \end{cases}, \tag{3.14}$$

where $p(x_t \mid z_{1:t-1}, u_{1:t-1}, m)$ is the posterior from the previous iteration, $p(z_t \mid x_t, m)$ is the measurement probability, which gives the probability of the measurement z_t given the state x_t, and $p(x_t \mid x_{t-1}, u_t, m)$ is called a state transition probability and corresponds to the motion model defined earlier. The prediction $p(x_t \mid z_{1:t-1}, u_{1:t}, m)$ is obtained by integrating the product of belief at time *t-1* and the probability that control u_t induces the transition from x_{t-1} to x_t.

In general the Bayesian filtering cannot be solved in a closed form. A popular method for Bayesian localisation is Monte Carlo Localisation (MCL), introduced Fox et.al [53]. MCL is a particle filter, which approximates the posterior distribution by a set of random samples drawn from it:

$$p(x_t \mid z_{1:t}, u_{1:t}) \sim \chi_t = \begin{bmatrix} (x_t^1, w_t^1) \\ \vdots \\ (x_t^N, w_t^N) \end{bmatrix} \tag{3.15}$$

A sample $\chi_t^i = (x_t^i, w_t^i)$ of a posterior distribution is called a particle. Each particle is an instantiation of the state at time t, with a weight that represents the probability of "how likely the particle is". The goal of the particle filter is to represent the posterior with the particle set. To do so, the number of particles should theoretically be infinite. In practice "a large enough number" is enough. The strong point of the particle filter is that it is non-parametric, and can represent practically any distribution. The downside is that it is a numerical approximation, which in many cases is computationally heavy.

The Bayesian filtering process is intuitively explained with MCL algorithm. If there is a posterior distribution χ_{t-1} and a dead reckoning estimate $(dx_t, dy_t, d\varphi_t)$ for movement $t-1 \rightarrow t$, the posterior at time t can be calculated using Algorithm 1.

Algorithm 1 goes through every particle. In Line 2a the prediction is calculated using the assumed pose at time *t-1*. The motion model $p(x_t \mid u_t, x_{t-1}^i, m)$ in this case calculates the motion from state (pose) x_{t-1}^i according to the input u_t and adds the necessary noise according to the specified noise model. For example, using the Equation 3.12 and assuming normally distributed noise with zero mean and σ^2 variance for the dead reckoning estimate, the noise is induced to the Equations as:

$$\begin{pmatrix} dx_t \\ dy_t \\ d\varphi_t \end{pmatrix} = \begin{pmatrix} dx_t^r + N(0, \sigma_x^2) \\ dy_t^r + N(0, \sigma_y^2) \\ d\varphi_t^r + N(0, \sigma_\varphi^2) \end{pmatrix}. \tag{3.16}$$

Now the Line 2a can be computed using Equations 3.7-3.10 directly. The result is that after this prediction each particle is moved according to the motion model to a new predicted state, with added noise, thus giving the transition from $p(x_{t-1} \mid z_{1:t-1}, u_{1:t-1})$ to $p(x_t \mid z_{1:t-1}, u_{1:t})$.

Line 2b in algorithm 1 incorporates the measurement into the distribution. The weight w_t^i is updated as a product of the previous weight, and the measurement likelihood is then normalised with η to force the distribution sum into one. Thus the weight can also be interpreted as a probability of the state (within given distribution).

To give an example, given a range measurement $z_t = r_m$ into a certain direction, the predicted position x_t^i and the map can be used to compute the predicted measurement r_p. r_p is the distance that would be returned by a sensor, which measures

Algorithm 1 Basic Monte Carlo Localisation Algorithm using dead reckoning

Inputs:

- map: m
- Dead reckoning information: $u_t = (dx_t, dy_t, d\varphi_t)$
- Pose distribution at time *t-1*: χ_{t-1}

1. $\chi_t^- = \chi_t = 0$
2. for i=1 to M do
   (a) sample $x_t^i \sim p(x_t \mid u_t, x_{t-1}^i, m)$
   (b) $w_t^i = w_{t-1}^i \eta p(z_t \mid x_t^i, m)$, where $\eta = \frac{1}{\sum w_t^i}$
   (c) add (x_t^i, w_t^i) into χ_t^-
3. endfor
4. if(Resampling)
   (a) for m=1 to M do
      i. draw i with probability w_t^i
      ii. add $(x_t^i, \frac{1}{M})$ to χ_t
   (b) endfor
5. else χ_t=χ_t^-
6. return χ_t

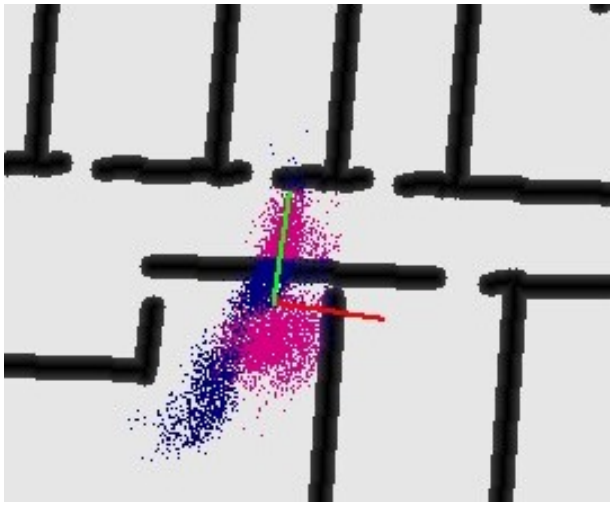


Figure 3.6: Posterior distribution with particle filter

with absolute accuracy, from pose x_t^i. Now $p(z_t \mid x_t, m)$ determines the likelihood of the real measurement being taken from x_t^i, given that the sensor and the map will have inaccuracies. If the inaccuracies are modeled as Gaussian, then $p(z_t \mid x_t, m)$ can be given as:

$$p(z_t \mid x_t, m) = \frac{1}{\sqrt{2\pi\sigma}} exp(-\frac{(r_p - r_m)^2}{\sigma^2}), \tag{3.17}$$

where σ is the standard deviation of the measurement inaccuracy. The weight is calculated for each particle, and the set of weighted particles approximate the posterior. An example of a pose posterior is illustrated in Figure 3.6. All the particle positions (x,y) are plotted in the figure with their colour-coded weights. The red particles have a good probability, while the blue ones have a low one. The distribution illustrated in 3.6 is a good example of the advantages of particle filters. The red and green coordinate axes in the image represent the weighted mean of the pose. The true pose of the robot in this case was in the corridor, which is the red area in the middle of the image. The posterior distribution in this case is called multi-modal.

If the loop between Lines 2 and 3 in Algorithm 1 were repetitively repeated, the particle filter would gradually diverge. This is because in Line 2a the prediction adds noise to the distribution and 2b only updates the weights, and thus it does not re-shape the distribution. The trick of the particle filter is in the resampling (Steps 4-5 in Algorithm 1). Intuitively, the resampling phase removes particles with a low probability and replaces them with copies of those that have a high probability. In this way a limited number of particles is resampled to the area which more probably represents the posterior of the estimate.

In practice the resampling discretises the estimated posterior. Therefore some care should be taken when the re-sampling is performed. Resampling too often might lead to a situation in which only a few particles represent the whole distribution, which

is not likely to be the true one. Sampling too rarely might cause the distribution to diverge. One indication of the distribution divergence is the variance of the weights. A large variance means that there are particles with very low and very high probabilities. This is the reason why Algorithm 1 has a conditional Sampling Importance Resampling (SIR) update. If no SIR is run the updated weight (Line 2b) is used; otherwise the weights are equalized to $\frac{1}{M}$.

Another strong argument for using particle filters is the flexibility with which a priori information can be added into it. Information such as "if the particle travels through the wall, then its weight should be zero", can quite easily be added to the calculation of the weights, but this would be very difficult with some other methods. The calculation of the likelihood is also very flexible. The information has to indicate how likely it is that the particle x_t^i represents the correct state. Additionally, $p(z_t \mid x_t, m)$ can fuse information from multiple sources. If the measurements are independent, the weighting can be computed as:

$$p(z_t \mid x_t, m) = \prod_{k=1}^{N} p(z_t^k \mid x_t, m). \tag{3.18}$$

3.2 Personal Navigation Systems

3.2.1 Overview

Personal Navigation is an inviting area as it has a wide area of applications and a large number of potential end users. Currently there is a lot of research on the area and many commercial products have been developed. The applications range from manufacturing (tracking assets and people) to single users (e.g. sports and location awareness services). Location-based services allow searches based on a person's location and, further, the location information may be used to guide the user to the location of a service. Personnel tracking may improve safety, security, and efficiency. Personnel tracking introduces beneficial applications, such as:

- doctors and vital instruments are found faster if the locations are known;
- patients tracking may trigger an alarm if a patient wanders;
- a location provides information for rescue personnel in the event of an accident (collapsed mine shaft, fire etc) [28, 41];
- in time-critical missions (e.g. SWAT, military, or fire-fighting) personnel tracking provides situational awareness [28].

Personal navigation is also widely used in sports. Wrist computers with GPS receivers, compasses, barometers, and pedometers are sold on the market. These

sports watches compute trajectories and profiles for the user, which provide additional value to their users. Personal localisation may also improve the quality of life by providing a means to aid visually impaired people in navigating in unknown environments [91].

In general, the solutions for personal navigation can be divided into three groups[141]:

- satellite-based systems
- network-based systems
- sensor-based systems

Satellite-based systems currently mostly use the well-known GPS (in future Galileo will also be used) and are mostly restricted to outdoor use. Network-based systems utilise existing Radio Frequency (RF) communication networks, such as GSM (cf. [22, 85, 108][22, 85, 108]), WLAN (cf. [141, 41, 24]), UWB (cf. [68, 140, 28]), or Bluetooth. The IEEE 802.11 wireless network is the most popular network for indoor positioning [141, 79, 41].

Sensor-based systems measure human movement with sensors that are carried by the user. The sensors measure the relative movement of a human by integrating accelerations, angular rates, and step and compass information. The sensor setups may also provide absolute information relative to the environment by measuring e.g. distances in the environment with ultrasound, infrared, or laser, or by using magnetic or barometric sensors. Sensor-based systems may be used to provide additional information to a system that fuses the information with e.g. a GPS or network-based system [91, 69, 141, 47].

Another way to divide the solutions is into outdoor and indoor applications. For outdoor use the GPS provides 30 satellites for accurate navigation. The European Galileo system will double the number of satellites, resulting in better coverage and making outdoor localisation even more accurate. There are works that fuse GPS with other means to get better coverage or accuracy (e.g. [91, 95]). In this thesis GPS is considered to be accurate enough to complement any personal navigation system that could be used indoors.

The more challenging areas for personal navigation are GPS-denied areas such as dense forests, mines, tunnels, and indoor environments (all of which will just be called indoors from now on). Indoor localisation methods can be divided into infrastructure- or beacon-based and stand-alone systems. The infrastructure-based methods rely on some existing or dynamically configured network of beacons and receivers. The stand-alone systems are used in the same way as the sensor-based systems above. These calculate the position of the person with respect to some starting position, without the need for any extra installations. The infrastructure-based methods are more often suitable for generic indoor localisation, while the stand-alone methods are designed for human localisation only. In many cases the methods are complementary, and one can benefit from another.

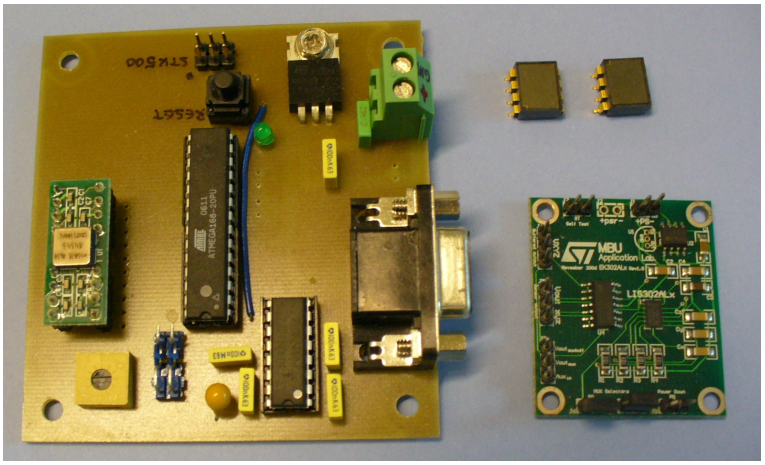


Figure 3.7: MEMS accelerometers and gyros

The following sections will review the current state of the art of indoor localisation. The emphasis is placed on the stand-alone systems.

3.2.2 Personal Dead Reckoning systems

Personal Dead Reckoning (PDR) systems estimate human motion by estimating the direction of movement, detecting steps, or measuring the size of the steps, the frequency of walking, the speed of the body, or the speed of the foot, or a combination of these. The following subsections will introduce the sensors and methods commonly used in PDR.

3.2.2.1 Sensors for Personal Dead Reckoning

PDR systems measure movement with body-mounted sensors, such as accelerometers, gyroscopes, magnetometers, and barometric pressure sensors. The development of Micro-Electro-Mechanical Systems (MEMS) technology has brought several small-sized, low-cost sensors for this purpose to the market. Typical MEMS accelerometers and gyroscopes are illustrated in Figure 3.7.

An accelerometer measures the linear acceleration of a body. The speed of the body may be calculated by integrating the acceleration signal once and the distance by double-integrating the signal over time.

Unfortunately, there are several factors that contribute noise to the acceleration measurement. Gravity causes errors in the signal if the orientation of the sensor is not perfectly known. To be able to integrate the distance of travel correctly one needs to have the sensor correctly oriented all the time (which does not hold true while walking). The output $a_m(t)$ of an accelerometer can be written as $a_m(t) = a(t) + a_{oe}(t) + a_d(t)$, where $a(t)$ is the true acceleration, $a_{oe}(t)$ is the orientation error and the $a_d(t)$ is the drift (e.g. temperature drift).Thus, double-integrating the

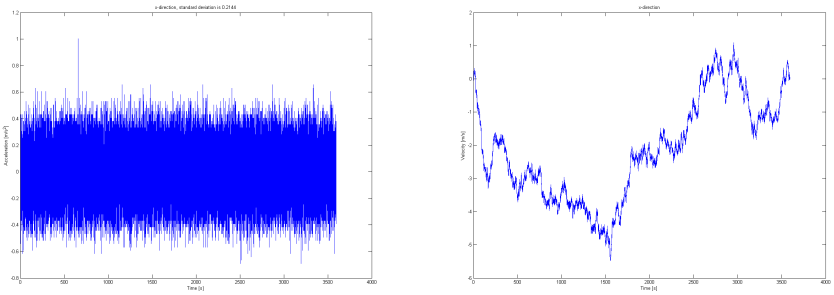


Figure 3.8: Drift-compensated acceleration signal (left) integrated into speed (right)

measurement will also integrate the bias components, resulting in a cumulative error. An example is given in Figure 3.8. The left-hand image shows raw acceleration measurement data recorded from a sensor that was static on a table for over an hour. The signal mean is subtracted from the signal to remove the orientation bias. The right-hand image shows the integrated velocity. Even though the measurement was made in optimal conditions, the speed value has a significant amplitude which exceeds 5 m/s.

Another displacement sensor used for PDR is a barometric pressure sensor. The atmospheric pressure changes with movement upwards or downwards and in principle provides an absolute measure of the height of the sensor. In this case the natural changes in the atmospheric pressure will cause bias in the measurement. Sagawa et al. [122] present a method for using a barometer as a vertical displacement sensor. They use a band pass filter which passes a signal around 0.3 Hz (with 1000 amplification). The design of the band pass filter is based on the assumption that the air pressure changes slowly, while electrical noise is rapid and walking up or down stairs is somewhere in the middle. The given transfer function for the filter is:

$$G(s) = \frac{-1000\omega^2 s}{s^2 + 2\zeta\omega + \omega^2}, \tag{3.19}$$

where the damping factor $\zeta = \frac{\sqrt{2}}{2}$ and the natural frequency $\omega = 0.6\pi$. The filter is a derivator and thus the vertical distance value is obtained by integrating the output of Equation 3.19 (and by multiplying by calibration coefficient). The error of vertical displacement was reported to be 11.1% [122].

A gyroscope measures the angular velocity around the measurement axis. Integrating the gyro output over time results in a heading estimation:

$$\varphi(t) = \varphi_0 + \int_{t_0}^{t} \omega(t)dt. \tag{3.20}$$

Similarly as in the accelerometer case, the measured output $\tilde{\omega}(t) = \omega(t) + \omega_b(t)$ is a sum of the bias and real angular velocity. Thus continuous integration will result in a continuously growing error in the heading estimate. One way of reducing the

error in the heading estimate is to estimate the bias using the average over some given time that the sensor is static:

$$\widetilde{b} = \frac{1}{N} \sum_{i=1}^{N} \widetilde{\omega}_i, \tag{3.21}$$

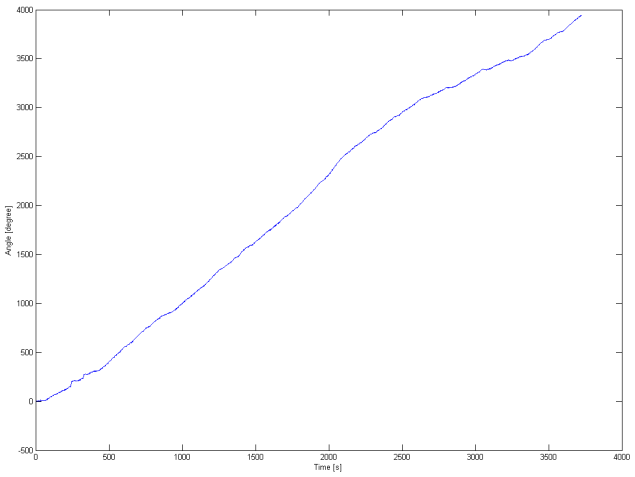
where ω_i is the sensor output sample when the sensor is static. While this works sufficiently well for high-quality gyros (such as fibre optic gyros or laser ring gyros), the method is not sufficient for low-cost gyros. Gyros introduce a term called bias stability. This refers to the change in bias measurement over time. An example of the sensor bias and its stability is given in Figure 3.9. The figures are obtained by reducing the bias from a complete data set using first 10 s, 100 s, and 1000 s for the bias estimation. It can be noticed that the bias calculated within the first ten or hundred seconds does not describe the complete bias. This is probably due to the temperature change inside the gyro during start-up. Even though the 1000-s bias averaging seems stable (rightmost image in Figure 3.9), within one hour the estimated angle has a maximum value of 450 degrees, which is more than one full circle.

The data were collected with a MEMS gyro. Other types of gyroscopes include the piezo electric gyro, fibre optic gyro, and ring laser gyro. MEMS and piezo gyros are usually the cheapest, but have an error of ~0.1 deg/s. Fibre optic gyros have an error of around 20 deg/hr and ring laser gyros of 0.1 deg/hr or less.

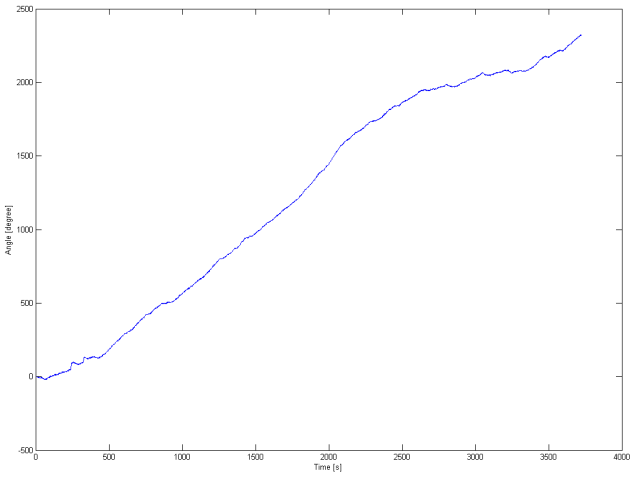
A magneto-resistive sensor can be used to measure the magnetic field of the earth and thus provide an absolute measure of the heading. These sensors are more familiarly called compasses. A compass is known to be sensitive to the magnetic disturbances that are caused, for example, by electric power lines and steel structures. Some of the disturbances may be removed with hard-iron calibration [132, 54]. The purpose of hard-iron calibration is to remove the disturbances caused by nearby static metallic objects by reshaping the output of the compass. Compass and gyro measurements are often fused to get a more reliable estimate of the heading. The gyros provide accurate short-term information, while a compass may have short-term biases, but long-term stability. A Kalman filter is a common way to fuse information. The estimation is done in two steps: 1) prediction and 2) updating. The prediction phase uses a system model and input to predict the state of the next measurement. A simple example is to use a gyro as the system input and have a constant system model; that is:

$$\begin{cases} \varphi(t|t-1) = \varphi(t-1|t-1) + \omega\Delta t \\ P(t|t-1) = P(t-1|t-1) + Q\Delta t \end{cases}, \tag{3.22}$$

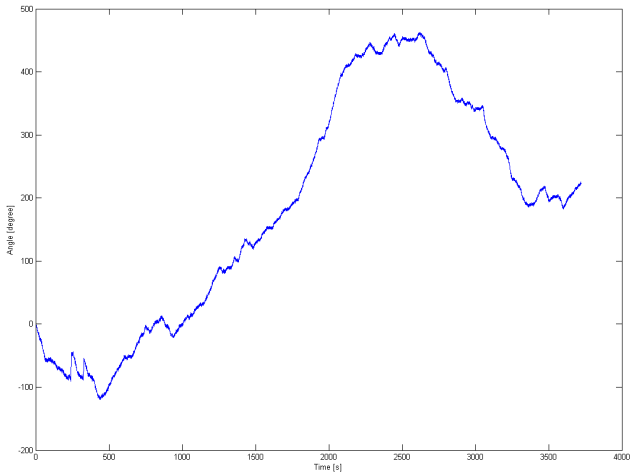
where $\varphi(t|t-1)$ is the predicted heading, $\varphi(t-1|t-1)$ is the estimated heading (during the last update), ω is the measurement of the angular speed within the time step Δt. $P(t|t-1)$, and $P(t|t)$ are the prediction covariance and posterior



1)



2)



3)

Figure 3.9: Integrated angle from gyro output using: 1) 10-s; 2) 100-s, and 3) 1000-s bias estimate. The sensor was static during the measuring.

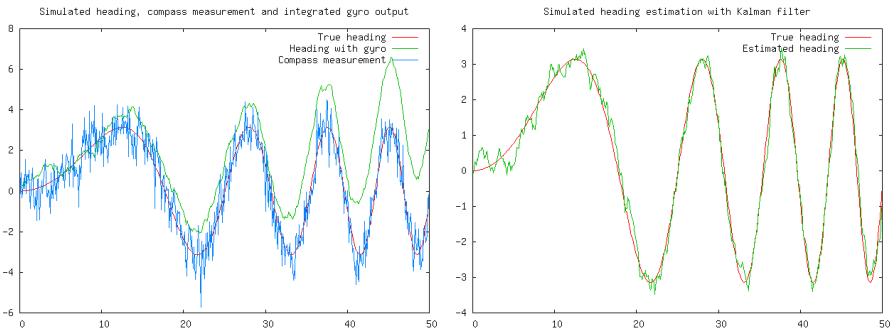


Figure 3.10: An example of Kalman-filtered heading using simulated data. In the left-hand image are the simulated measurements and true heading. On the right are the estimated output and the true heading.

covariance, respectively. Q is the variance of the gyro measurement. An updated heading estimate is calculated when the measurement z is got from the compass:

$$\begin{cases} e = z - \varphi(t|t-1) \\ S = P(t|t-1) + R \\ K = P(t|t-1)S^{-1} \\ P(t|t) = P(t|t-1) - KSK \\ \varphi(t|t) = \varphi(t|t-1) + Ke \end{cases}, \tag{3.23}$$

where R is the compass variance and K is so-called Kalman gain.

An example of the estimation process is given in Figure 3.10. The left-hand image in Figure 3.10 hows a noisy compass measurement and a biased gyro measurement with the true heading. The estimated heading is shown in the right-hand image, with the true heading.

The Kalman gain is adapted in Kalman filtering, so that the prediction and update leads to an optimal solution (in terms of minimum variance). The Q and R parameters have a great effect on the K and thus they are the tuning parameters. If greater reliance is being placed on the gyro, then Q is small and R should be big and vice versa. The difficulty in choosing the parameters comes from the fact that these should describe the true error of the sensor. A compass may provide sub-degree accuracy in optimal conditions. However, in indoor conditions disturbances may cause the sensor bias to be in tens of degrees. The bias from the environment is unpredictable and can cause a coloured heading estimate.

Kim et al. [77] present a method that detects the compass disturbance by comparing the compass and gyro angular rates to detect if there is any magnetic disturbance. The method works if the disturbance is sudden, but not if the disturbance is constant, for example, if the user is standing still in front of an electric power line. Foxlin [54] states that the disturbances from the environment are correlated to the position. He proposes that compass measurements should be used only every now

Figure 3.11: Collection of Inertial Measurement Units

and then (e.g. only after a new step has been taken). In this way the measurement errors are less correlated with each other and can be treated as additional noise.

Inertial Measurement Unit (IMU) is an integrated 3D-measuring sensor often composed of three accelerometers and three gyroscopes. Some of the sensors are equipped with additional magnetometers. The IMU configuration is called a "strap-down configuration" if the components are fixed into a common chassis and are not actively controlled [97]. This is almost always the case with low-cost MEMS IMUs.

An IMU is needed to estimate the position and attitude of a free-flying object. In Figure 3.11 shows a few currently available MEMS IMUs. The size of these components has been scaled down and they are now cubes only a few centimetres square, which makes them feasible for human dead reckoning too.

Attitude and position tracking is an iterative process that is often referred as strap-down inertial navigation [97, 104]. All values are measured in a sensor coordinate system with respect to the sensor body and therefore should be converted into the world coordinate system for subsequent processing. The transformation of the accelerations into the world coordinate system requires knowledge about the orientation of the sensor.

The sensor orientation (α =Roll, β =Pitch, γ = Yaw, see Figure 3.12) can be described by the transformation matrix E between the initial orientation (initial frame) of the sensor and its current orientation. At the beginning E is an identity matrix, or it may be initialised using the initial knowledge of the sensor orientation. The matrix E is updated by partial rotations (so-called small angles) during the measurement with respect to rotation around all the axes of the sensor in each measuring step. The small angles are used as parameters for the rotations and are computed as the integrated angular velocity (independently in each axis) between two successive measured data samples.

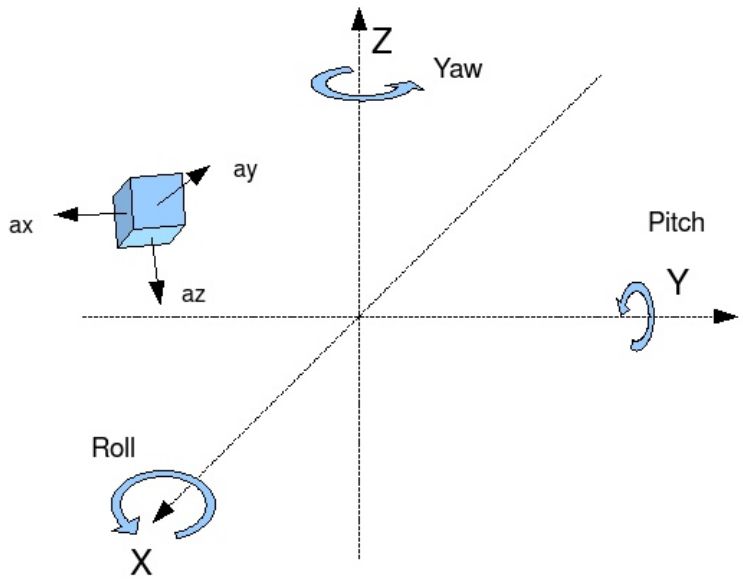


Figure 3.12: World frame of reference and sensor coordinates

In this manner the updating of the matrix E can be described as an iterative process:

$$E(t) = E(t-1)R_x(\Delta\alpha(t))R_y(\Delta\beta(t))R_z(\Delta\gamma(t)), \tag{3.24}$$

where $\Delta\alpha(t), \Delta\beta(t), \Delta\gamma(t)$ are integrated angle values from the sample *t-1* to t. The rotation matrices are defined as:

$$R_x(\alpha) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & cos(\alpha) & -sin(\alpha) \\ 0 & sin(\alpha) & cos(\alpha) \end{bmatrix} \tag{3.25}$$

$$R_y(\beta) = \begin{bmatrix} cos(\beta) & 0 & sin(\beta) \\ 0 & 1 & 0 \\ -sin(\beta) & 0 & cos(\beta) \end{bmatrix} \tag{3.26}$$

$$R_z(\gamma) = \begin{bmatrix} cos(\gamma) & -sin(\gamma) & 0 \\ sin(\gamma) & cos(\gamma) & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.27}$$

The orientation of the sensor is used for the transformation of the acceleration values from the sensor coordinate system to the world coordinate system as follows:

$$a_w(t) = E(t)a_m(t), \tag{3.28}$$

where a_w is the acceleration vector in the world frame of reference and a_m is the measured acceleration vector in sensor coordinates. After the measurement is transformed to the world coordinate frame, the effect of gravity (z-component with a

Figure 3.13: Stance phases

magnitude of g) must be subtracted from a_w. The transformed accelerations and angular velocities can be integrated to update the velocity and position estimates in the world frame of reference.

Using an IMU provides a means to estimate a position in 3D. However, IMU is no better in the sense of drift and biases than a single gyro or an accelerometer. The error sources in this case are the drifts in angular velocities and the error in the initial pose of the sensor. After transformation, the gravitation is removed from the z-axis and the acceleration values are integrated into velocities or positions (2nd integration). Additional errors in this transformation come from the drifts in accelerations.

3.2.2.2 Human motion analysis

To achieve greater accuracy in the estimated position, further assumptions must be made. Practically all PDR systems take advantage of how humans move. In general, there are nearly unlimited possibilities for human motion (e.g. Wikipedia introduces some 26 different "gaits"); however, the average human moves by means of walking or running. Walking is a cyclical process, which has two main phases: a stance phase (the foot is on the ground) and a swing phase (the foot is in the air) [4].

The stance phase can be further divided into the initial contact, loading response, mid-stance, and terminal stance phases. The swing phases are likewise divided into the preswing, initial swing, midswing, and terminal swing phases [4]. Figure 3.13 illustrates the stance phases: the loading response (left); mid-stance (centre), and terminal stance (right) phases. According to [4] the loading response phase is about 10% of the gait. During the loading response phase the foot comes into full contact with the floor and the whole weight of the body is on the stance foot.

A simple experiment was performed to get an idea of the dynamics of human motion. The data were collected with an IMU (Microstrain Inertia-Link) that was mounted: 1) on the foot; 2) at the waist, and 3) on the head of a walker. Figure 3.14 shows 3D accelerometer measurements taken from one foot while walking. In the figure Line 1(red) is the forward acceleration (roughly, since the accelerometer was not perpendicularly mounted on the foot), Line 2 (green) shows the sideways acceleration, and Line 3 (blue) the vertical acceleration. The left-hand image shows the clear pattern between the stance phases and swing phases. The right-hand image

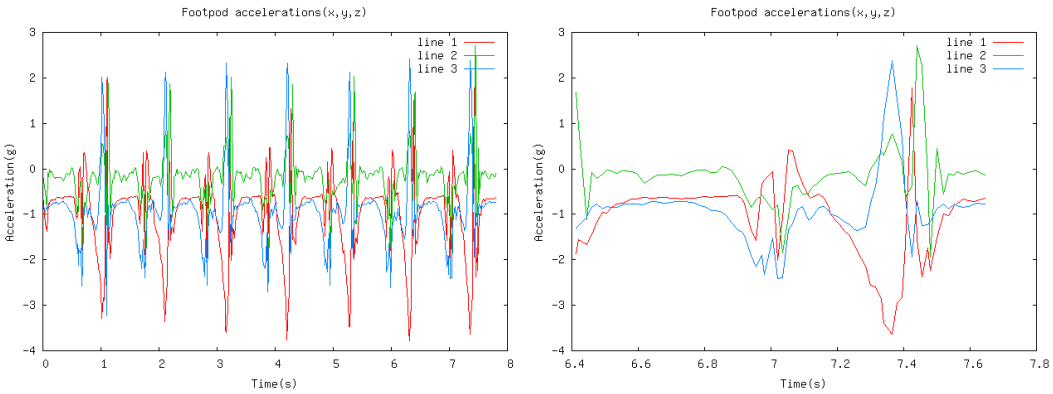


Figure 3.14: Accelerometer signals from one foot while walking. Left: continuous walking, right: one gait cycle.

shows one gait cycle. The signal shows that between 6.5s and 6.8s the leg is in the mid-stance. During this time there are no accelerations applied to the sensor. In the terminal stance preswing phase (around 7s) the leg rotates, which is visible as oscillations in the acceleration signal. The toe-off is followed by swing phases which are visible in long forward acceleration (Line 1: from 7.1s to 7.4s), which ends in the initial contact and loading response. The initial contact can be read from a high acceleration peak of vertical acceleration (Line 3) and from the rapid deceleration of forward acceleration. This is the moment when the heel hits the ground and it is often used for step detection.

The forward velocity of the body is near-constant while walking. At the same time the foot velocity varies from zero to more than double the body velocity. Additionally, the vertical trajectory of the centre of mass is smoothed by the coordinated motion of the foot, ankle, and knee (see Figure 3.15). For sensor placement this establishes some principle differences. In the earlier work the division of sensor placement can mostly be made between body-mounted sensors [46, 141, 91, 112, 84, 47, 77] and foot-mounted sensors [54, 104, 122, 8, 132]. Additionally, in [7] a helmet-mounted PDR is presented and in [72] a handheld device (although the device is not in the hand but e.g. in a pocket).

Figure 3.16 shows a comparison between forward acceleration signals measured from the foot, waist, and head while walking. The foot signal has the largest variation. It is also relatively easy to read the different stages of walking from this signal. The magnitude of the signals measured from the waist and head is lower and there is no stance phase visible, which supports the fact that the body is moving close to a constant speed. The right-hand image in Figure 3.16 plots the vertical accelerations from the same sensor placements. Again the accelerations measured from the foot have the greatest variance, but now the vertical accelerations from the waist and head show a "wave" pattern. This pattern is caused by the vertical movement of the centre of mass, as illustrated in Figure 3.15.

A good deal of work has been done on analysing human walking in the personal dead reckoning context, as well as in many other applications. For example, Kato

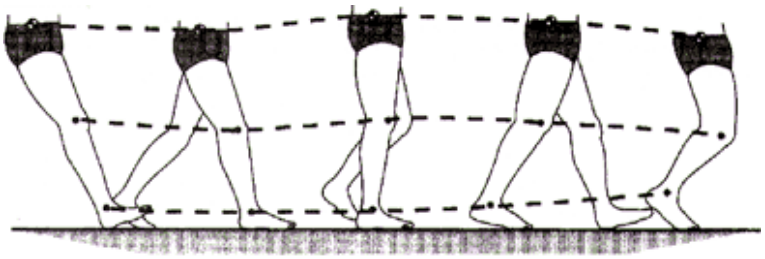


Figure 3.15: Coordinated motion of foot, knee, and ankle smooths the pathway of center of mass (courtesy of [4])

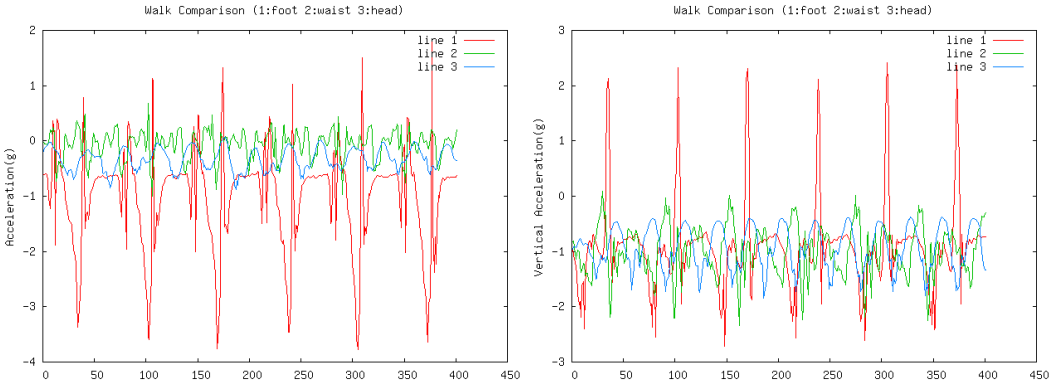


Figure 3.16: Acceleration signals measured from foot, waist, and head. Left: forward accelerations, right: vertical accelerations.

[73] presents early work on human stride measurement. The stride is measured with an ultrasonic transmitter/receiver pair attached to the shoes. The device measures the stride continuously. The results show that while walking there is a dominant leg and a subordinate leg, which means that the stride length may differ between the left and right feet. It was also observed that the stride length varies during the acceleration phase, which was especially visible in the fast walking experiments. Usual gait times of 1.5s for slow walking, 1.2s for normal walking, and 1.0s for fast walking were reported.

Ladetto analysed the variation in step length in [84]. He experimented with 20 persons who walked with a constant frequency. He obtained results that indicate that with a given frequency the step length varies from 15% (60 steps/min) to 4% (130 steps/min). From the same tests he concluded that there is a correlation between the step length and step frequency. For example, a mean length of 60 cm was obtained with 60 steps/min and one of 90 cm was obtained with 130 steps/min.

3.2.2.3 Step detection

One way to determine human motion is to detect the gait of the human and to estimate the length of the step instead of directly estimating the movement of the sensor. In this way the error of the sensor noise is not integrated over time. Instead, the error accumulates as a function of the number of steps taken. There are two terms used in the literature to describe the measurement of human gait: step length and stride length. Step length is the distance between successive foot strikes of the opposite foot. Stride length is the distance between successive foot strikes of the same foot [132, 4].

Basically, all PDR systems detect human gait somehow. The body-mounted systems usually detect the peaks (maximum or minimum detection) of the vertical acceleration (heel impacts) [84, 47], or use the zero-crossing of the acceleration magnitude signal [72, 88, 141]. The zero-crossing method requires a three-axis accelerometer. The method is based on the wave-like movement of the human body. The implementation of the method is shown in Algorithm 2. An example of the output of Algorithm 2 is shown in Figure 3.17. The green spikes show the moment of the step detection (zero-crossing). In this case the signal from an IMU is used directly but often the signal is said to be filtered with a low-pass filter to obtain more robust results [72, 84, 141].

Kim et.al. [77] present a step detection algorithm that continuously matches the foot movement to the gait pattern. The step is considered to be detected only if the swing and load response phases occur in the right order. The detection of the phases is based on thresholding the acceleration signal to lower and upper limits that are based on which phase of the gait is ongoing.

Step detection with foot-mounted sensors is usually performed by detecting the stance phase of the walking [54, 105, 104, 122, 8]. The basic idea is to detect from the signals a moment when the signal is close enough to zero (mid-stance). Ojeda

Algorithm 2 Acceleration Magnitude Zero-Crossing step detection

1. Compute acceleration magnitude: $A = \sqrt{a_x^2 + a_y^2 + a_z^2}$
2. Subtract the magnitude of gravity: $\tilde{A} = A - g$
3. Compute the sign of signal $\tilde{A}$: $S = sgn(\tilde{A})$
4. The steps may be counted from the zero-crossing i.e. when S changes sign (e.g. from negative to positive)

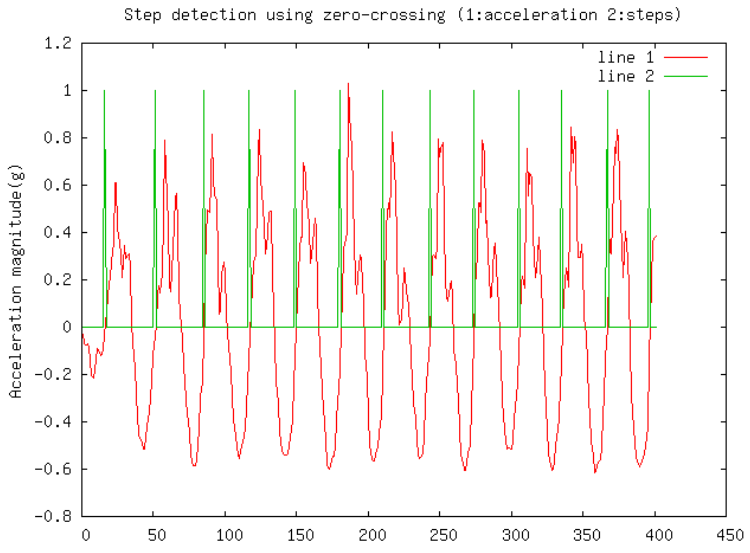


Figure 3.17: Step detection result using zero-crossing method

and Borenstein [104] describe a step detection algorithm that uses the magnitude of three-axis gyro values, with experimentally tuned thresholding to detect the "zero-velocity" of the foot. Beauregard [8] describes a similar approach, but he used the product of acceleration magnitude and angular rate magnitude.

3.2.2.4 Step and stride length estimation

The simplest pedometers estimate the distance by detecting steps and use a constant length for the step. Obviously this is not accurate for position estimation, but can be satisfactory for estimating energy consumption or the distance travelled during the day (for example, for health purposes). For position estimation, there are two categories of methods used for step length estimation: direct and indirect. The direct methods use the sensor data directly to estimate the step (e.g. by double-integrating the accelerations of the foot), while the indirect methods estimate the step from sensor data that are only evidence of the step length (e.g. measuring only the vertical acceleration while walking).

Weinberg [142] presentes an example of indirect step length (l_{step}) estimation using vertical acceleration:

$$l_{step} = \sqrt[4]{A_{max} - A_{min}} * C_0, \tag{3.29}$$

where A_{max} and A_{min} are the maximum and minimum accelerations measured during a single step and C_0 is a calibration constant. The formula is based on the "bounce" movement of the hip while walking. This method has been used at least in [141, 47]. According to [47] Equation 3.29 provides step length estimate within $\pm 3\%$ for the same subject and $\pm 8\%$ across the variety of subjects.

Ladetto [84] studies step length estimation using a body-mounted IMU. He estimates the step length using the acceleration in the direction of movement (antero-posterior). In his work he shows that there is a strong correlation between the step length and step frequency and presents Equation 3.30 for the estimation.

$$l_{step} = C_0 + C_1 * f_{step} + C_2 * var(a_m(t)), \tag{3.30}$$

where C_0, C_1, and C_2 are parameters calculated by linear regression, f_{step} is the measured step frequency, and $var(a_m(t))$ is variance of acceleration signal. On the basis of the natural characteristics of step length, he does not use Equation 3.30 directly. Instead he proposes the following procedure:

1. consider a constant step value l_{step} at an interval of N steps;
2. compute the residuals between l_{step} and the predicted value using Equation 3.30;
3. estimate the mean and the variance of residuals $N \sim (\mu, \sigma^2)$;

4. compute the the next constant $l_{step} = l_{step} + \mu$.

This approach assumes that on average the length of a human step varies around a stable value. It also smooths the effect of possible outliers. Ladetto also considers step length adaptation using Kalman Filtering. He incorporates the GPS information using:

$$\frac{D_{GPS}}{N_{step}} = l_{step}, \tag{3.31}$$

where D_{GPS} is the distance traveled (according to GPS) during N_{step} steps. Ladetto reports an accuracy of less than 2% of distance traveled.

Another indirect method is to integrate the absolute value of acceleration magnitude [72, 77] over the step. For example, Kim et al. [77] estimate the stride length by integrating the mean of the forward acceleration:

$$l_{stride} = 0.98 \sqrt[3]{\frac{\sum_{k=1}^{N} |a_m(k)|}{N}}, \tag{3.32}$$

where the N is the number of acceleration samples during a stride.

A neural network has also been utilised in step length estimation [23, 7]. In principle, linear regression and a neural network are the same in the sense that in both cases the parameters are fitted to given data. The results are also similar. Cho et al. [23] report the distance error to be about 2% on average and Beauregard [7] reports an error of "a few percent".

The direct estimation of step or stride length basically requires sensors to be placed into the foot. Kato built an ultrasound-based device to measure the length of steps [73]. The device constantly measures the distance between the transmitter and receiver on the basis of Time-Of-Flight. The measurement provides a direct step length estimate. Yeh et al. [146] present a similar approach to Kato's system. The distance between the legs is calculated using modified Cricket electronics [111]. The foot placement is additionally estimated with accelerometers, which provides complementary information. However, the paper reports large errors for both methods (of a magnitude of 20% or more of the distance travelled).

A more conventional way to measure steps directly is to track the foot movement during the swing phases. The basic principle is to estimate the position of the foot in 3D. In essence this approach uses the Inertial Navigation formulae (such as 3.24 - 3.28), but instead of the estimate being calculated constantly, the estimation is performed during the swing and corrected during the stance phase.

Sagawa [122] presents an early approach which uses three-dimensional accelerometers and a gyroscope to track the pitch of the foot. The estimation is not performed in full 6-DOF, but instead only in the direction of travel. With this method the paper reports an error of up to 5% of the distance walked. Similar work is presented

by Stirling et al. [132] with two 2-DOF accelerometers and a compass triad. In their work the orientation of the sensor was also used to estimate the movement of the foot in a plane. The actual stride length is computed by estimating the mean stride velocity and the actual stride time.

Foxlin [54] is probably the first to use 6-DOF IMU to estimate foot movements in 3D. The measurement is performed with a foot-mounted wireless IMU, which provides 3D acceleration, a 3D angular rate, and 3D magnetometers. Foxlin presents a Kalman Filter-based solution, which estimates the position, pose, and drifts of the sensor. The stance phase is used as a pseudo-measurement. During the stance phase the foot is still, which means that the measurements should be zero and the foot velocity should be zero. This information is used to update the sensor bias terms and to correct the speed estimate, which have accumulated errors. This process is called Zero-Velocity Update (ZUPT). This method is not just an estimate of the step length. It actually estimates the position of the foot in 3D, which gives the position, height, and orientation of the foot directly, without extra sensors. Almost identical work is presented by Ojeda and Borenstein [104, 105] and later by Beauregard [8], with the difference that in these the estimation is performed using strap-down inertial navigation algorithms directly. In all cases the overall accuracy is reported to be around 2%.

3.2.3 Bounded error Personal Navigation Systems for Indoors

Personal Dead Reckoning systems provide information about movement with respect to some initial conditions. As described above, the errors are typically from 2%-10% of the distance travelled. In position estimation, the heading error will also have a significant value. For example if a gyro drifts 1 degree within 100 m it can cause an error in position of 0.86 m. The same happens if a compass gives a 5-degree error within 10 m. This means that no matter how good the PDR system is, the error will eventually grow to be unbounded. To limit the error one needs to have a measurement that is based on some reference.

In [46, 141] a fusion of WLAN-based fingerprinting, a map, and a PDR system with a particle filters is presented. The PDR system presented by Evennou and Marx [46] is based on step detection, assuming a constant step length. The heading was estimated with a gyro. Wang et al. [141] additionally estimated the step length using Equation 3.29. In both studies basically the same likelihood function for particle updating is presented:

$$p(z_t \mid x_t, m) = \begin{cases} 0, \; if\; particle\; crosses\; wall \\ \frac{1}{\sqrt{2\pi\sigma}} \exp(-\frac{r_e^2}{\sigma^2}), \; otherwise \end{cases} , \tag{3.33}$$

where r_e is the distance between the position indicated by the particle and the measured position, and σ is the standard deviation of the measurement.

A system based on the Cricket system [111] and inertial navigation is presented by Popa et al. [110]. In this work the PDR approach was used to provide the tracking,

Figure 3.18: Dead Reckoning Module from PointResearch Corp

while the Line-Of-Sight was lost with the Cricket system. Herrera et al. [66] present initial results with an integrated UWB and PDR system. In both studies the results that are shown are only for a very limited indoor area.

Another approach is described by Korougi et al. [81], who use RFID tags as location information sources. The tags are detected within 1.5 m and so the "cell ID" method can be used when the location of the tags is known. The tags are situated in places of interest (e.g. in front of paintings) and thus they provide a mean to reset the dead reckoning errors. Further work by Kourogi is presented in [80]. The localisation system is based on an IMU and a wearable camera. In the calibration phase images from known locations are recorded into the database. The absolute reference measurement is obtained by the image registration of the pre-recorded images and the measured ones. Kourogi's approach requires a teaching phase, but does not rely on any infrastructure. This is a major difference to the other works presented. All the other systems require some infrastructure to be available and calibrated.

3.2.4 Experimenting with commercial systems

3.2.4.1 DRM - III Dead reckoning module for personnel positioning

The DRM-III (Dead Reckoning Module) is a module for human positioning manufactured by the Point Research Corp. (see Figure 3.18). The module is designed to be belt-mounted and provides the position information through a serial port. The accuracy of the module is promised to be from 2% to 5% of the distance travelled. The DRM module can also incorporate GPS, where available.

The DRM measures the displacement of the user from the initialisation by measuring the direction and the distance travelled with each footstep. The heading is measured using an electronic compass. The distance is measured by tracking each footstep

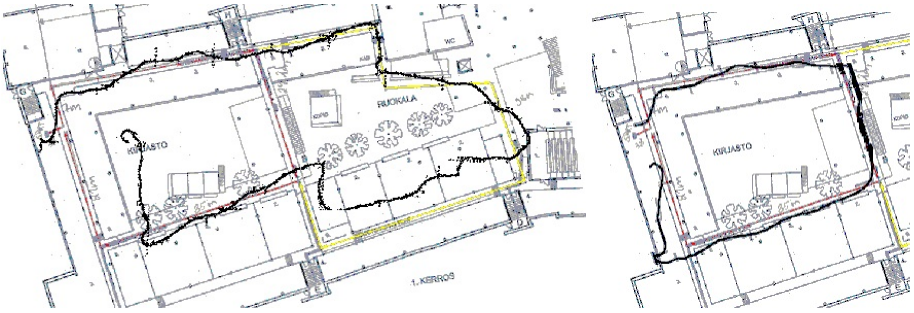


Figure 3.19: DRM-III test runs: the left-hand route is approx. 220 m long and the right-hand route is approx. 100 m. long. The path was enclosed.

using accelerometer data. The preliminary tests showed that the module relies heavily on initialised stride length. The module estimates the stride taken to be close to the initial value; even the length of the step would substantially differ from the initial value. If the initial stride length is set correctly, under normal walking and running conditions the average error is within 10% of the distance travelled (see fig 3.19).

Moreover, the module was found to fail when the subject was walking backwards or sideways. The DRM-III can be used to provide a rough position estimate for a short time. However, it does not provide that consistent accuracy which would make it suitable for use for stand-alone dead reckoning estimation to be used with other methods.

3.2.4.2 Polar S3

The Polar S3 is a foot pod sensor for measuring the distance travelled by a runner. S3 is used with Polar's wrist computer (Figure 3.20). The sensor provides an average step length and cadence (frequency) and distance estimates. It requires calibration, which is performed by running a known distance and giving a correction coefficient for the wrist computer. The sensor is designed mainly for running. Nevertheless, it was tested so that the sensor was calibrated to normal running, and then tested for different gaits.

The sensor provides the distance travelled with an accuracy to within 10 m. To make the measurement error reasonable, 200 m was used as a reference run in all the tests. Figure 3.21 shows the results of the tests. The graph shows the estimated speed (blue bars) during different runs. The number on top of these bars gives the number of the test set. Additionally, the sensor provides information about the average step length (the green line at the top of graph), as well as about the frequency of steps. The wrist computer reports the distance travelled along the path, which was manually recorded; the results of each run can be found in Table 3.1.

When the test subject ran at the speed that the sensor was calibrated to (the first eight runs), the sensor provided exactly 200 m as a result. When the subject was walking (quickly and normally (10-14)), the sensor provided an error of 10 m (5%).

Figure 3.20: Polar S3 and Wrist computer

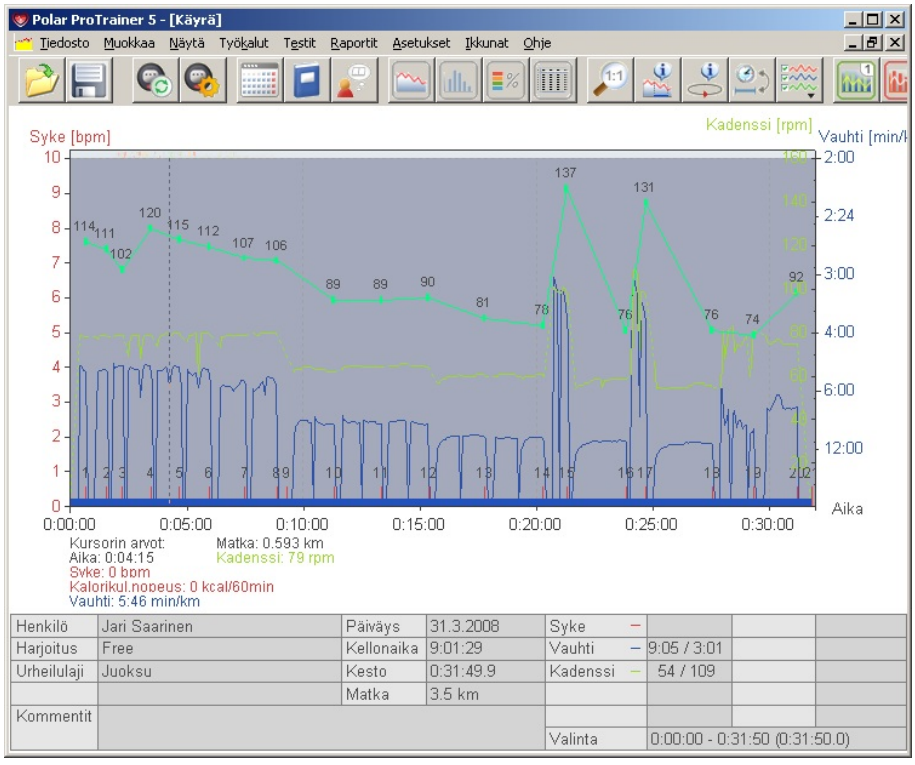


Figure 3.21: Results with Polar S3

Run	Distance	Meas	Err(m)
Normal run	200	200	0
Normal run	200	200	0
Normal run	200	200	0
Normal run	200	200	0
Normal run	200	200	0
Normal run	200	200	0
Quick walk	200	210	-10
Quick walk	200	210	-10
Quick walk	200	210	-10
Normal walk	200	210	-10
Normal walk	200	210	-10
Slow Walk	200	210	-10
Slow Walk	200	220	-20
Quick run	200	180	20
Quick run	200	180	20
Obscure walk	200	170	30

Table 3.1: Test runs with Polar S3

A quick run and a very slow walk resulted in an error of 20 m (10%). Figure 3.21 also shows the variation in the stride length. In the graph the lowest stride length is 74 cm for the very slow walk and 137 cm for the fastest run.

Overall the S3 provides a good estimate of the distance travelled in various conditions. Unfortunately, the sensor cannot be used as a sensor for position estimation, mainly because the system is not designed for it (it is a closed system which provides length updates only every now and then). However, it shows that a foot-mounted sensor can provide accurate distance information.

3.3 Laser-based Localisation

Personal dead reckoning has been widely researched and it has been shown that it can be used as a complementary method to infrastructure-based methods. It is evident that personal dead reckoning cannot be used alone as a long-term solution for personal navigation.

In this thesis the target is the development of a personal navigation system that is able to provide a long-term position estimate with a bounded error. One of the main design criteria is that the system has to be able to localise a human without external infrastructure. A stand-alone system has the advantage that it can be used in all buildings and conditions, regardless of the infrastructure available. For example, in a fire-fighting situation at least the infrastructure inside the building can be damaged and would therefore be unusable.

One approach which has not been studied this far in the context of personal navigation is the use of environmental perception sensors. In mobile robotics there have been a large number of studies on localising a robot with just onboard sensors. The methods are based on matching the environment perception data to a given model (map) or on the robot building the map from the sensor data while localising itself. The main objective of this section is to review the traditional (robotic) localisation methods and try to adapt the methods for human indoor positioning. Furthermore, emphasis is placed on laser-based localisation methods.

3.3.1 Review of range sensors

An environment perception (also sometimes called exteroceptive) sensor measures the actual state of the environment. The most common sensors measure distances to obstacles (reflection from obstacles) or provide visual images from the environment. Probably the most traditional range measurement sensor is sonar (Figure 3.22). Sonar is a time-of-flight sensor, and is based on ultrasound. The typical range accuracy of sonar is usually some centimetres (up to a few metres), but the angle resolution is some tens of degrees. Sonar is usually used with occupancy grid maps. Sonar has been successfully used as a localisation sensor in [137, 139]. Tardos [137] successfully demonstrates a mapping and localisation procedure using a stochastic feature map. Thrun et al. [139] use a sensor for localisation in the context of particle filtering. Sonar is good for detecting obstacles near the robot. The wide sound "beam" allows the sensor to detect even very narrow obstacles (e.g. chair legs). Another advantage of sonar is that it is insensitive to lighting conditions (or even to smoke) and it can detect glass. The disadvantages are that sonar is inaccurate and cannot be used to measure distant obstacles.

Another traditional type of ranging sensor is radar. Radar works by sending radio pulses and by displaying the echoes on the basis of their time of arrival. There are a few localisation applications that use radar. Probably the best known is the work done by Dissanayake et al. [31]. In this work the radar was used to provide measurements to radar reflectors to test Simultaneous Localisation and Mapping (SLAM) methods. Radar-based localisation was later used in an autonomous straddle carrier system [38].

Scanning laser range finders (also called lidars) have become popular devices for mobile robot localisation (see Figure 3.22). The most common sensor in the past was the SICK LMS series sensor, which is able to measure a 180-degree field of view with an angular resolution of 0.5 degrees. The sensor provides an accuracy of approx 2 cm up to 80 m. One sweep of measurements provides an accurate 2D range image, which can be used in various ways to build a map for localisation (e.g. [61, 139, 5, 60, 126, 59] to mention a few). SICK provides accurate measurements up to a long distance, but it is heavy (the indoor version weighs approx 5 kg) and consumes a lot of power. Recently, the Hokuyo company from Japan has developed a small-sized sensor called the URG-X002, with low power consumption for indoor

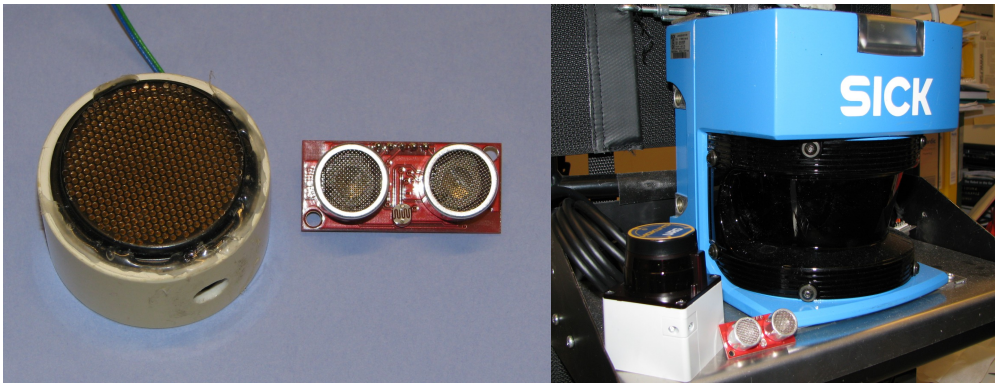


Figure 3.22: Range sensors. Left: Ultrasonic sensors, Right: Lidars

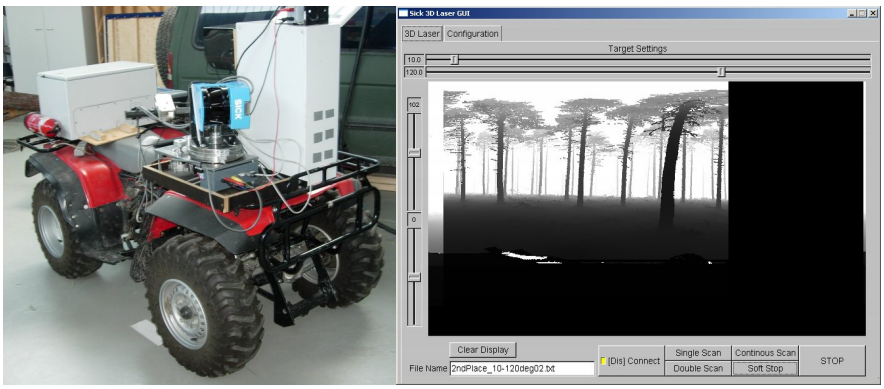


Figure 3.23: 3D Laser setup with SICK LMS 200. [128]

conditions [75]. The sensor is only about 5 cm in width, weighs 170 g, and is able to provide measurements up to 4 metres.

Recently, there has been a movement from 2D environment perception to 3D perception. One solution is to use 2D lidars and equip them with an additional rotation axis to provide measurements in 3D [101, 128]. In figure 3.23 (left) shows one such system, which was developed at Helsinki University of Technology. The right-hand image shows the raw range measurement taken by the device. The weight of the system is over 10 kg and it is therefore not suitable for small robots or for personal navigation.

A new wave of 3D sensors is the so-called 3D cameras, which directly measure 3D depth images. The 3D camera is based on illuminating the environment and by measuring the reflected light with a grid of detectors. As a result the sensors provide high-frequency 3D measurements of the environment. The first prototypes have already been sold, and some initial results have been reported [114, 92]. While these sensors are interesting and small enough for personal navigation, they are still sensitive to various conditions and thus the first mapping applications with these sensors are waiting for the "next generation" [92].

A camera is another type of sensor altogether for the perception of the environment. While a range sensor provides a distance or set of distances and angles to surrounding

obstacles directly, a camera provides a high-resolution image of the environment. One image does not provide any information about the distances to obstacles; instead it provides the shape of the objects as reflected light intensity, colour etc. The distance information can be derived by using stereo cameras or by estimating (or knowing) the movement of the camera between a set of images. Plenty of work has been done on the camera navigation of mobile robots [29]. A camera can provide rich data on the environment with cheap and readily available technology. However, in this work camera navigation is ruled out.

3.3.2 Laser scan matching

Scan matching is a process of calculating differential movement on the basis of consecutive range scans. The problem is to find a 2D transformation which maximally overlaps the scan taken at time t (called *a current scan*) with a scan that was taken at time $t-1$ (called *the reference scan*). The resulting 2D transformation (rotation R_φ and translation T) is given with respect to the reference scan and is effectively the same as described as $(dx, dy, d\varphi)$ in Section 3.1.2.1 (see also Figure 3.4a). Because of this, the scan matching is also referred to as laser dead reckoning [5] or laser odometry [130].

The scan matching procedure is illustrated in Figure 3.24. A range finder travels through the environment and at time $t-1$, the reference scan s_{t-1} is taken (indicated by red crosses in Figure 3.24a). A range scan is a set of range-angle pairs $s_{t-1} = \left[(r_0^{t-1}, \theta_0^{t-1}), \ldots, (r_N^{t-1}, \theta_N^{t-1})\right]$ taken in the sensor coordinate system. In this case the sensor has reported one range measurement (the reflection from the nearest obstacle in the given direction) every one degree, resulting $N = 180$ and $\theta \sim [-\pi/2, \pi/2]$. At time t a new measurement (the *current scan*) s_t is taken, which is indicated in figure 3.24a as cyan.

In Figure 3.24b both scans are transformed into (x,y)-plane with respect to the sensor coordinates (i.e. both frames are expected to be initial frames) using equation 3.34 for both scans.

$$m_t = \begin{bmatrix} r_0^t cos(\theta_0^t) & r_0^t sin(\theta_0^t) \\ \vdots & \vdots \\ r_N^t cos(\theta_N^t) & r_N^t sin(\theta_N^t) \end{bmatrix}^T \tag{3.34}$$

The primary assumptions in scan matching are that: 1) the scans are taken from the same environment, and 2) the scans are close enough for there to be enough overlapping points in both scans. The goal of the matching is to find a transformation which represents the points in m_t in the coordinate system of the reference scan m_{t-1} so that there is a maximal number of overlapping points. The transformation converts the points of the current scan into new coordinates defined by Equation 3.35.

$$\tilde{m}_t = R_{d\varphi} m_t + T, \tag{3.35}$$

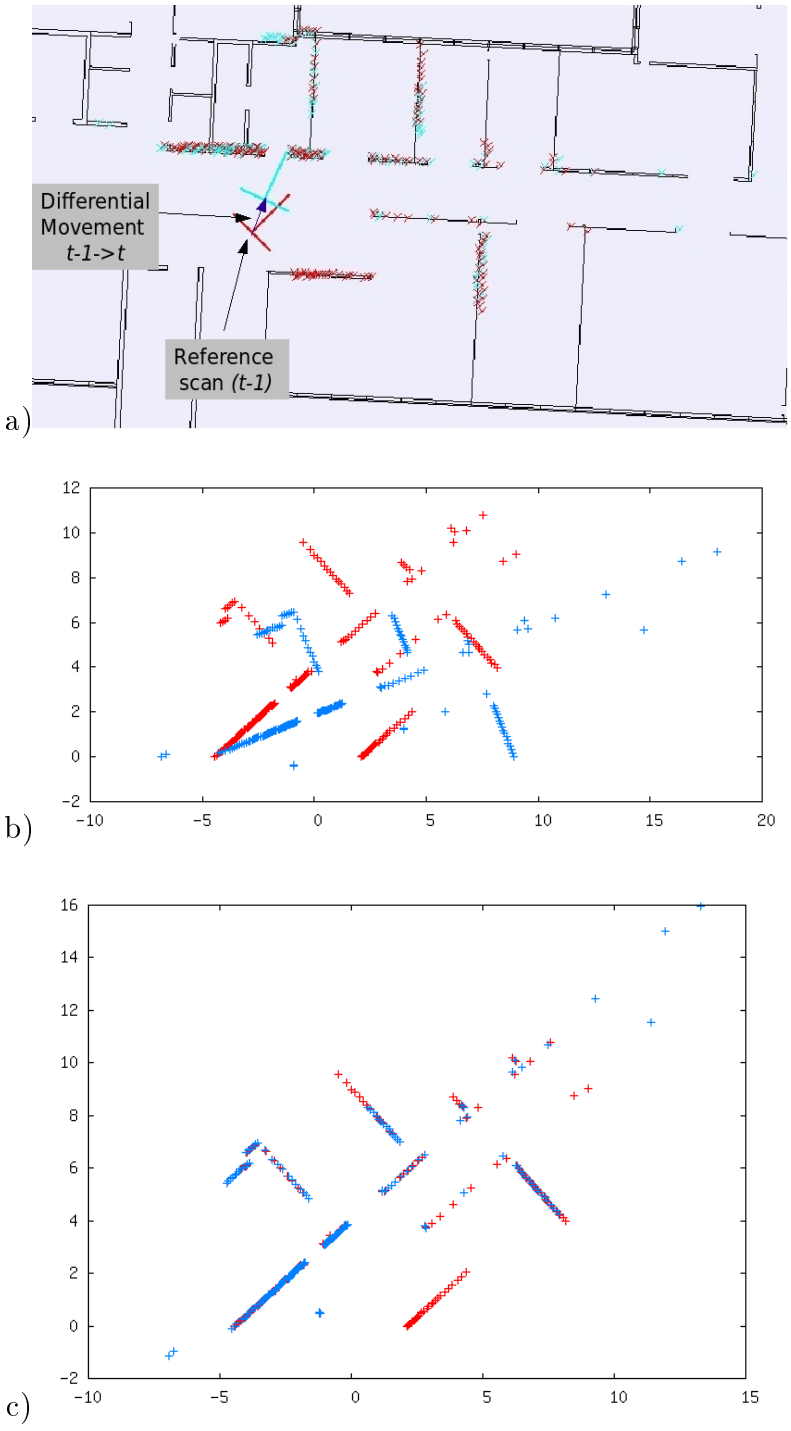


Figure 3.24: Scan-matching procedure example: a) scans projected into global frame of reference; b) scans are plotted in laser coordinate system; c) scans projected with respect to reference scan.

where $T = \begin{bmatrix} dx \\ dy \end{bmatrix}$ is the differential translation between the frames and $R_{d\varphi}$ is the rotation matrix:

$$R_{d\varphi} = \begin{bmatrix} cos(d\varphi) & -sin(d\varphi) \\ sin(d\varphi) & cos(d\varphi) \end{bmatrix}, \tag{3.36}$$

and $(dx, dy, d\varphi)$ is the differential movement $t-1 \rightarrow t$. The result is illustrated in Figure 3.24c.

It is noteworthy that not all the points are taken from the "same environment". For example, Figure 3.24c shows that there are plenty of points that are only visible in one scan. Additionally, the scan points are not necessarily taken from exactly the same position in both scans. The angular resolution of 1 degree means that the "spacing" between the scans is a function of the distance, resulting in sparse measurements at long distances. Furthermore, the sensor noise causes deviation in the measurements, even those would be taken from the same position in space.

Laser dead reckoning and traditional dead reckoning can often complement each other. If both have been estimated for the same time interval, the result should be the same. Normal dead reckoning can have errors which are caused, for example, by wheel slippage or inaccuracies in the motion model, to which laser dead reckoning is not sensitive. However, laser dead reckoning is sensitive to cases in which the environment is repetitive (or symmetric) or does not have a sufficient amount of features or to dynamic variations in the environment. A practical way is to use the normal dead reckoning information as an initial guess for the scan matching and to calculate only the correction from the scans. Having a good initial guess for the scan matching simplifies the search for a correspondence between the scans and usually makes it more robust and faster. Additionally, the independent dead reckoning estimate can be used to detect failures in scan matching.

There are three basic approaches to scan matching:

1. search in feature space
2. search in pose space
3. translation-invariant transformations with cross-correlation

The first approach searches for the same features (such as points or lines) from both scans. The transformation is calculated according to the feature pairs that are found. The second method searches the pose space (or solution space) to find the value that gives the best correlation between the scans. The third approach separates the estimation of heading and translation by transforming the scans into a function that is invariant to translation.

Searching in the feature space requires a correspondence to be found between the features in the two consecutive scans. On the basis of such a found correspondence an

error function is created, which is minimised to obtain the rotation and translation parameters. The different methods are often categorised into point-to-point, point-to-line, and line-to-line methods [93, 61]. Point-to-point methods search directly for a point-wise correspondence between the scans. The points may also be features that can be represented as points (corners, cylinders etc). Point-to-line methods extract the lines in the reference scan and correlate the points in the current measurement to the line features (usually the shortest distance). Line-to-line methods extract the lines from both scans and find line segments that represent the same polygonal object.

The Iterative Closest Point (ICP) algorithm is probably the most famous scan-matching algorithm. The algorithm was introduced by Besl and McKay [9] for shape registration in 3D. ICP is a point-to-point algorithm, which uses the minimum Euclidean distance as the criterion for correspondence. Lu and Milios [90] apply ICP to 2D range scans and present two additional variations called Iterative Matching Range Point (IMRP) and Iterative Dual Correspondence (IDC) algorithms. IMRP searches for the matching range values from both scans (which must be approximately from the same bearing) to determine the point pairs. IDC is a combination of the two. The rotation and translation parameters are obtained by minimising the squared error of the N matching point pairs; that is [90]:

$$E_{dist}(d\varphi, T) = \sum_{i=1}^{N} \| R_{d\varphi} m_t + T - m_{t-1} \|, \tag{3.37}$$

where m_t are the points in the current scan that match points m_{t-1} in the reference scan, $R_{d\varphi}$ is the rotation matrix and T is the differential translation between the frames. Equation 3.37 can be solved in closed form using the least squares method. Given that all matching points are correct, Algorithm 3 calculates the solution in one iteration. In reality this is not the case and that is why the algorithms are iterative. The basic ICP algorithm transforms the current scan little by little using the result from Algorithm 3 and then repeats until the sum of squared distance error between the point pairs is small enough. Basically, ICP methods work well if the movement between scans is sufficiently small. However, if the displacements are large the algorithm may converge into a local minimum.

Diosi and Kleeman [30] present similar work to IMRP, but instead of the correspondence being sought in Cartesian coordinates, the pairs are searched for in polar coordinates. They take advantage of the fact that the scanner range values are ordered, which simplifies the search and, according to them, makes it faster.

Cox [26] presents early work on map-based localisation using using a laser range finder. He uses a line map as a model and matches the points returned from the scanner to the map. The same method can be used for scan matching (implemented e.g. in [61]). The advantage of finding a correspondence to a line is that the correspondence can be calculated as the minimum distance to the line instead of to a single point. As mentioned before, the scan points are not necessarily taken from the same position.

Algorithm 3 Least Squares Minimization for point correspondences

Input: N matching point pairs (m_r, m_c), where $m_r = \begin{bmatrix} x_r^1 & y_r^1 \\ \vdots & \vdots \\ x_r^N & y_r^N \end{bmatrix}$ and $m_c = \begin{bmatrix} x_c^1 & y_c^1 \\ \vdots & \vdots \\ x_c^N & y_c^N \end{bmatrix}$ are point in reference scan and current scan.

1. Compute averages for both sets: $\begin{cases} \bar{x}_r = \frac{1}{N}\sum_{i=1}^{N} x_r^i \\ \bar{y}_r = \frac{1}{N}\sum_{i=1}^{N} y_r^i \end{cases}$ and $\begin{cases} \bar{x}_c = \frac{1}{N}\sum_{i=1}^{N} x_c^i \\ \bar{y}_c = \frac{1}{N}\sum_{i=1}^{N} y_c^i \end{cases}$

2. Compute covariance: $\begin{cases} S_{xx} = \sum_{i=1}^{N}(x_r^i - \bar{x}_r)(x_c^i - \bar{x}_c) \\ S_{yy} = \sum_{i=1}^{N}(y_r^i - \bar{y}_r)(y_c^i - \bar{y}_c) \\ S_{xy} = \sum_{i=1}^{N}(x_r^i - \bar{x}_r)(y_c^i - \bar{y}_c) \\ S_{yx} = \sum_{i=1}^{N}(y_r^i - \bar{y}_r)(x_c^i - \bar{x}_c) \end{cases}$

3. Calculate transformation: $\begin{cases} d\varphi = atan(\frac{S_{xy} - S_{yx}}{S_{xx} + S_{yy}}) \\ dx = \bar{x}_c - (\bar{x}_r cos(d\varphi) - \bar{y}_r sin(d\varphi)) \\ dy = \bar{y}_c - (\bar{x}_r sin(d\varphi) + \bar{y}_r cos(d\varphi)) \end{cases}$

Line-to-Line ([93, 62, 20]) or, more generally, feature-to-feature methods ([89, 5]) extract features from both scans and try to find a correspondence between the features. The number of features extracted is usually significantly lower than the number of scan points, which makes them faster than point-to-point methods. On the other hand it is crucial to find the correct correspondence between the features. Because there are only a few features, a wrong association has a much more significant influence than in point-to-point methods. The conventional correspondence search is to compare individual features and to select the ones that are close enough in both scans to be pairs (e.g. [93]). A more sophisticated method is to acknowledge that the features are correlated from scan to scan (i.e. it is likely that there is a set of features in both scans that are the same) [98, 62, 5]. For example, Bailey [5] forms a graph from the features in both scans and searches for the maximum overlap between the graphs to find the correct correspondences.

The weakness of feature-based methods is their sensitivity in the absence of features. This is in general a problem of all methods which use features (also point-to-line ones). For example, Lingemann et al. [89] acknowledge that there are situations in which their feature extraction fails to give enough features (such as when approaching a corner). This was acknowledged by Gutmann and Schlegel [61] in their work and they ended up proposing a method that combines IDC and the matcher proposed by Cox [26].

Correlation-based methods [127, 130, 10] search for the solution in the pose space directly. All correlation-based methods have the following steps:

1. pick a pose from the search space;
2. transform the current scan accordingly;
3. calculate the score using *a correlation function.*

The correlation function shows the goodness of the given pose. The correlation function does not require individual points to be associated or features to be identified, but it should have the property of giving the maximum (or minimum) with the correct translation and rotation parameters.

The methods differ in the way the pose space is searched through and what kind of correlation function is used. Schultz and Adams [126] used correlation to match a local evidence grid to a global one. The method is not directly scan matching as they used several scans and odometry to build the local map, but the idea also applies to scan matching. Schultz and Adams present two correlation functions, a binary match and a product match. A binary match sets the score for a cell as one if it is occupied (or is not occupied) in both the local and global grids. A product match calculates the score as the product of the evidence in both grids. The total correlation for a single pose is calculated as a sum of scores. They also present two alternative searchers for the pose space searching. An iterated hill climber initially divides the pose space into cells. The search is performed around the expected pose

and if a better score is found, the searcher moves into the neighbouring cell and repeats the search. If a better score is not received, the resolution is doubled around the expected cell. The procedure repeats until the desired accuracy is reached. A centre-of-mass searcher divides the search space into initial cells, but this time a random sample is taken from the cells and then evaluated. The result is obtained in the end by calculating a weighted centre of mass on the basis of score values. The paper reports no significant difference between the correlation functions, but the centre-of-mass searcher was generally found to be better.

Biber [10] uses a method called Normal Distribution Transform (NDT) for scan matching. NDT calculates an approximation of normal distribution to discretised cells on the basis of the scan points that are in the cell. The normal distribution then shows "the probability of measuring a sample from some position within the cell". Biber uses the transformation directly to obtain a score for matching.

The third class of methods differs in that instead of an attempt being made to find the rotational and translational parameters in a coupled manner, the estimation is separated into the estimation of individual components. One way to achieve this is to transform the scans into a form that is invariant to translation. The basic assumption is that given two scans s_{t-1} and s_t that are taken from the environment, the tangent direction of the obstacles remains the same in both scans. Basically, line-to-line methods naturally take advantage of this, because if the line pairs are associated, the heading difference can be computed as the (weighted) average of the differences between the line headings [93]. Weiss and Puttkamer [143] (and later Röfer [113]) propose a method that uses a histogram and cross-correlation for scan matching. The first step is to compute an angle histogram of both scans. Given that the range scans are ordered, the histogram can be computed by discretising the result of Equation 3.38 for all scan points.

$$\alpha_i = arctan(\frac{y_i - y_{i-N}}{x_i - x_{i-N}}), \tag{3.38}$$

where N is a fixed number used to smooth the measurement noise. As a result, the histograms are discrete value functions, which are the same, but with a phase shift that is:

$$h'(i) = h(i + j), \tag{3.39}$$

where $h'(i)$ is the histogram function of the reference scan and $h(i + j)$ is the histogram function of the current scan. The phase shift is directly the heading difference between the scans and can be found by searching for the $'j'$ that maximises the cross-correlation function [143]:

$$Cor(j) = \sum_{i=1}^{n} h'(i)h(i + j). \tag{3.40}$$

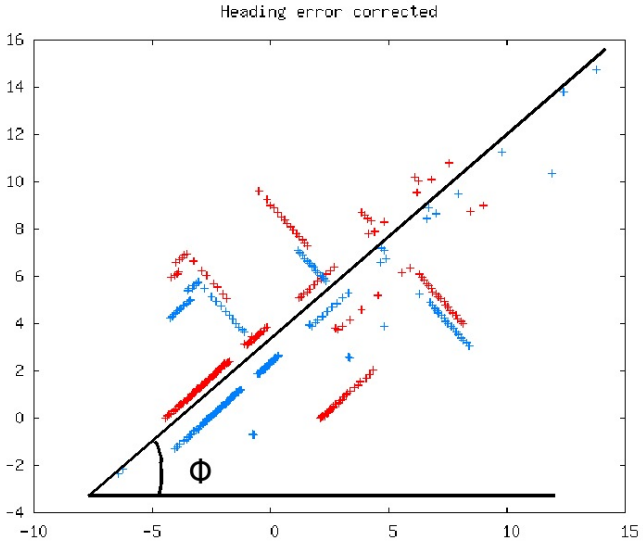


Figure 3.25: Common direction in the histogram matching

In polygonal environments the histogram and its cross-correlation can be used to find the translational shift too. After the correct rotations between the scans have been found, the current scan can be rotated to the same heading as the reference scan. The histograms for x- and y-translation are calculated with respect to "common directions". The common direction is illustrated in Figure 3.25 and can be found by searching for the maximum of the histogram [143]. The common direction is such that it aligns the scan so that a maximum number of scan points represents a single x (or y) value.

The work of Censi et al. [21] falls into the same category, but instead of using histogram transformation they use Hough transformation. Hough transformation also provides invariance to the translation, and the orientation can be found in a similar way to the histogram case, which can be used to search for translations. Both methods are dense methods in the sense that they use the whole data set as such.

3.3.3 Map-based localisation

In the previous section the reference for localisation was derived from the previous measurement. In this section the current measurement is compared against a known static map. The goal is to find a correspondence between the map and the measurement and derive either a correction to the current estimate (the continuous localisation case) or find an absolute location within the map (global localisation).

Basically, all scan matching algorithms can be converted into map-based localisation. The point-to-line [26] and line-to-line [62] methods can naturally be used with line maps (instead of a correspondence to lines derived from the reference scan being searched for, the correspondence is searched for from the model). The work done

by Schultz and Adams [126] fuses a "batch" of measurements as a local grid to a global grid. The global grid in their work was incrementally built, but the method could be used just as well with a known map. A general way to use scan-matching algorithms for map matching is to compute the expected scan from the expected pose and use that as a reference scan.

Continuous localisation is an easy task using the above methods, given an initial pose, a perfect model (map), a static environment, a good estimate of movement (dead reckoning), and perfect measurement. Unfortunately, none of these is usually the case. Uncertainty is always present in real localisation, which is why localisation has been formulated in a probabilistic manner for decades[1]

A popular approach has been (and is) to use the Extended Kalman Filter (EKF) for pose tracking. EKF provides a minimum variance estimate of the state on the basis of a motion model and a measurement model. EKF uses a uni-modal distribution (represented by mean and variance) to track the pose. Intuitively, the motion transfers the mean and adds variance, while the measurement reduces the variance. Both the measurement and the map also have noise, which causes the result of matching the measurement to the map to be treated as a distribution as well (i.e. the measurement can be taken from an area defined by a mean and variance). The method has been used since the late '80s and has been demonstrated to work with artificial beacons (e.g. in the Automatic Guided Vehicle (AGV) application [14, 39] and later in the autonomous straddle carrier system [38]), as well as with the natural geometric beacons [87].

The scan-matching methods presented above (or any methods that can produce a pose based on matching a measurement to a map) can be used with EKF. For example, Sciele and Crowley [124] present similar work to that presented by Schultz and Adams, but they used EKF for tracking the pose. Additionally, Gutmann et al. [63] use scan matching and EKF for tracking the pose.

EKF is a solid, efficient, and well-formulated method for continuous localisation, as long as everything goes as expected. The justifications for not choosing EKF are: 1) it cannot solve the global localisation problem, and 2) EKF can only represent uni-modal pose distributions. The first refers to a case in which the robot initially starts from an unknown pose within a known environment (or the pose is suddenly lost for some reason). The second is perhaps even more critical. Uni-modality means that given a time instant the pose is represented with a mean of the pose and its variance. However, there are several possibilities of ending in a situation in which the pose can be just one of the many possibilities (represented by a number of mean-variance pairs, cf. Figure 3.6). In the mid-'90s Burgard et al. [16] proposed probability grids as an alternative approach to solving Bayesian localisation. The idea of a probability grid is to represent the posterior of the pose belief over all the possible poses in a discretised manner. In other words, the probability grid represents all the discrete values that the robot can be within the given environment. The probability grid approach is able to represent multi-modal distribution and it can be used for

[1] The formulation of the probabilistic continuous localisation was made in section 3.1.2.2.

global localisation. However, the method is computationally heavy, requiring plenty of memory. To represent a belief of a 2D pose, the probability grid is a 3D grid, with Nx*Ny*Na components. The number of components depends on the size of the environment and the desired resolution of the poses (e.g. Konolige and Chou used a 38-m x 30-m map with 10-cm and 2-deg resolution, which adds up to 20×10^6 poses). The prediction and update step must be calculated for each pose cell, which makes the algorithm extremely heavy in large environments. Konolige and Chou [78] used correlation to approximate the measurement likelihood. The algorithm was more efficient than the maximum likelihood version, but the number of poses still remains the same.

The most influential method for this thesis is the one called Monte Carlo Localisation (MCL) presented by Fox et al. [53][2]. MCL approximates the beliefs by using samples and basically the integration is performed in a Monte Carlo fashion [64]. MCL possesses the same advantages as grid localisation, but the main difference is that the belief is approximated with samples that are concentrated around the most probable states. The advantage of MCL over a grid-based approach is that it can represent the posterior more accurately, with fewer computational requirements. For more details on MCL see Section 3.1.2.2, where it was already introduced.

[2]See also section 3.1.2.2

Chapter 4

Novel Methods for Sensor-Based Personal Navigation

4.1 Overview

The major motivation for this thesis has been the PeLoTe project, which was a proof-of-concept study of a search and rescue concept based on cooperating human and robotic entities. The reference mission was targeted to a large-scale office environment. This set some restrictions for the personal navigation system:

- the mission happens indoors;
- preliminary installation cannot be assumed;
- preliminary installations/calibrations inside the building cannot be made;
- the map of the building is not necessarily correct.

This chapter presents localisation methods that satisfies above assumptions using self-contained sensor system called PeNa. The goal is to integrate PDR methods (see section 3.2.2) with methods used in robotics (see section 3.3) to obtain bounded error localisation system (see section 3.2.3) that uses only a map as a reference. The overall localisation system is presented in Figure 4.1. The personal dead reckoning part is based on continuous step length estimation, combined with compass and gyro information for heading estimation. The integrated values are given to laser dead reckoning for fine-tuning of the pose estimate and finally, the pose is referenced against the map. The map is assumed to be known, or built by robotic members of the team.

The schematic in Figure 4.1 shows the localizstion process used in the final demonstrations (section 5.3). However, the location information can be derived using various combinations of different methods. For example, the map-matching block in Figure 4.1 requires a dead reckoning input, which can be obtained from either of

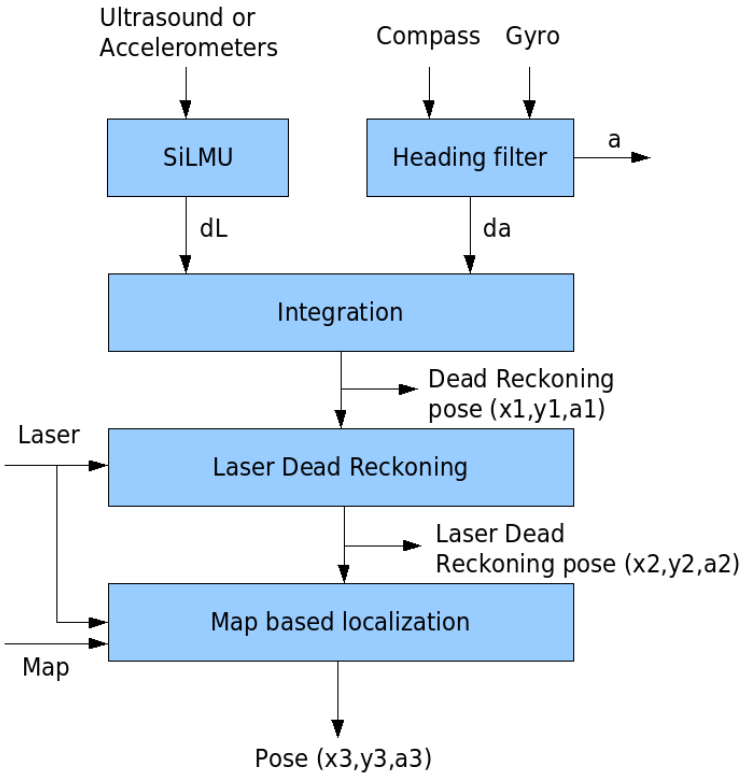


Figure 4.1: Schematics of the overall localisation procedure

the dead reckoning blocks. In the schematic the heading can be estimated using a heading filter, which provides an absolute heading, or as a differential change in the heading between the scans. Additionally, later in this chapter a method for map matching will be presented that is based on odometry and map matching, without lasers being used at all. Furthermore, laser dead reckoning does not necessarily require any input of initial movement.

In principle, the methods presented provide a means to build a personal navigation system that: 1) uses personal dead reckoning and a map, without any environment perception sensors; 2) uses a laser as the only sensor, or 3) uses a combination of all of these. In practice the third option was selected for use in the final version because it was found to be the most robust at the time. The results for the second option are not presented in detail.

4.2 Test equipment

PeNa is a portable sensor system, built around a standard hiking backpack (see Figure 4.2). PeNa was developed to support the development of personal navigation and to demonstrate the Personal Assistance System (PAS) in the human-robot team demonstration. Figure 4.2 shows the system that was used in the final demonstrations.

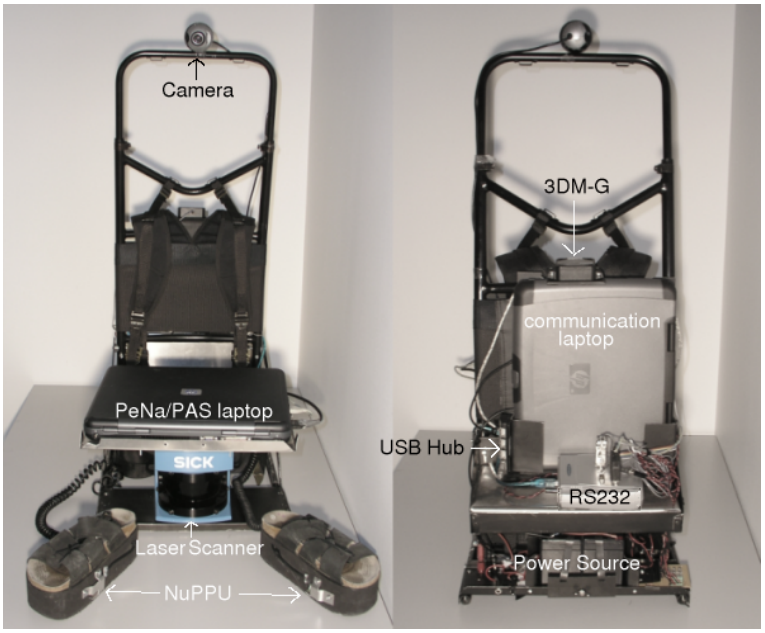


Figure 4.2: Hardware of the Personal Navigation System

The PeNa hardware includes batteries, power converters, a step length measurement unit (called SiLMU or NUPPU, which are introduced in Section 4.3.2), a fibre optic gyro, a 3DM-G IMU, a compass, a SICK LMS200 laser scanner, a camera, and two laptops. One laptop is installed in front, and makes all the necessary calculations and serves as a display for the user interface. The other laptop is reserved for communication purposes (voice, images, and data).

The solid frame supports the equipment and makes the load easier to carry. The total weight of the system is approximately 14 kg without the laptops.

For the personal dead reckoning special boots were designed. During the research two different approaches for step length estimation were used: one is based on ultrasound and the other is based on accelerometers. Both systems were mounted on the shoes, providing a continuous estimate of the foot movement. The ultrasound-based system provided a continuous measure of the distance between the feet. The accelerometer-based system was selected for the final demonstrations, because there were other systems (namely robots) using ultrasound devices and they might have caused interference. Both systems will be introduced in detail in Section 4.3.

For the heading estimation PeNa provides a Hitachi Fibre Optic Gyroscope HOFG-X, a compass, and a MEMS IMU from Microstrain (3DM-G). The 3DM-G is a 3D orientation module, with 3D magnetometers, 3D gyros, and 3D accelerometers. HOFG-X provides good short-term stability compared to 3DM-G, but is significantly more expensive. The HOFG-X sensor was used in the heading estimation 4.3.1, which was used when deriving the laser dead reckoning results (section 5.2.2) for the algorithm presented in section 4.3.4.1.

The environment perception sensor is a standard SICK laser range finder. The selection was based on the availability of such a sensor. The sensor is bulky and

heavy and consumes too much power to be taken as a serious sensor candidate for real personal navigation purposes. Nevertheless, the sensor provides a means to study if a 2D range sensor could be used for personal navigation [1].

4.3 Personal Dead Reckoning

The task of the personal dead reckoning in PeNa is to provide the system input for the other localisation modules as shown in Figure 4.1. The output is used as an initial estimate of the movement either for laser dead reckoning or for the map matching. This is important because: a) the laser-based methods are computationally heavy, and b) the error sources in the laser measurements will be less likely to cause errors if the initial guess is close to correct.

For continuous localisation the estimate should be continuous and it should limit the error. The PDR estimate should at least be better than a zero movement estimate. While this requirement may seem trivial, it is not. Many step detection systems are unable to detect the direction of a step. A step taken backwards or sideways may be considered as a step taken forward. In this case the estimate given by the PDR system is actually worse than the zero movement estimate. Two alternatives for step length estimation are presented in this section: 1) ultrasound-based, and 2) accelerometer-based systems. In both cases the estimation is performed with foot-mounted sensor systems. In both cases it is expected that the main gait will be walking.

There are many good reasons why the two systems should be integrated. The ultrasound-based system measures the step length directly, while in the accelerometer case the step length has to be estimated indirectly. The ultrasound-based system is unable to detect the direction of the step, and if it is used alone this can violate the "better than zero movement estimate" principle if the user does not walk as expected. The accelerometer-based system, on the other hand, is able to detect the step direction. Unfortunately, the two systems have never "co-existed", which is a reason for presenting them separately in this thesis. Nevertheless, to develop an ultimate PDR system was not the key target for this thesis. Instead, the PDR system is required by other methods presented later.

4.3.1 Heading Estimation

The heading estimation can be used in two ways: 1) as an estimate of absolute heading based on magnetic sensors and gyroscopes, or 2) as a change of heading between two time instants. The first one is an attractive option, as it would always give the absolute orientation regardless of the other localisation means. However,

[1] Currently, there are already sensors available that could be used to replace the SICK sensor, and could be considered as candidate as "a serious sensor candidate for real personal navigation purposes" (for example the latest release from Hokuyo UTM-30LX sensor).

it is a well-known fact that indoors the electric fields and steel structures cause disturbances to the magnetic field (in our tests up to ± 40 degrees).

To test the feasibility a basic version of the heading estimator was implemented using the Kalman filter (Equations 3.22 and 3.23). HOFG-X was used to provide the angular rate input. The typical drift of this gyro is approximately 2 deg/min, including also the additional "drifts" caused by the movement of the rotation axis during the walking process. To tune the Kalman filter properly, one needs to find a proper Q/R ratio (the ratio between the noise in the compass and gyro in this case). Too much reliability on the compass causes the filtered value to follow the compass too much. On the other hand, too little reliability causes a drift in long-term use.

The major problem in the heading estimation is that the compass noise is coloured. This means that in some locations the values of the compass are permanently biased. Foxlin [54] proposed using the compass measurements only if there has been enough movement between the measurements to reduce the effect of biased measurements. The problem that was encountered was that in a modern office building, where the system was tested, there are a significant number of electrical wires on the ceilings of the corridors, which was visible as a bias that was evident for several tens of metres. Figure 4.3 (top) shows the measurements from the gyro and the compass and the resulting heading estimation. The bottom image illustrates the location dependency of the measurements. In Figure 4.3 there are several locations where the compass measurement is strongly biased (e.g. straight paths between Locations 1 and 2). As a result the Kalman filter was tuned to follow the gyro (making the whole estimation process practically useless).

Because of this and to get rid of the expensive (and not generally available) fibre optic gyro, the decision was taken to use the gyro output of the 3DM-G IMU module to estimate the change in the heading between the given time instants.

4.3.2 Step Length Estimation

The Step Length Measurement Unit (SiLMU) measures the distances between the ankles. The distance measure is based on the time of flight of the ultrasound signal between a transmitter and a receiver. The transmitter and a reflector are attached to the right foot and the signal detection module is on the left foot. To be able to measure the distance continuously, the transmitted signal must be audible to the receiver. This requires a beam angle of almost 180 degrees for the sound. One solution is to use multiple transmitters and receivers placed around the shoes. Another solution is to use reflectors to spread the sound beam of the transmitter to a half-plane (see Figure 4.4).

The length measurement is performed continuously at a rate of 60 Hz by a microcontroller. from which the value is requested by the CPU. The consecutive distance measurements form a pattern that represents a sine wave with a DC offset. The measurements from the microcontroller are filtered to reduce outliers from the data.

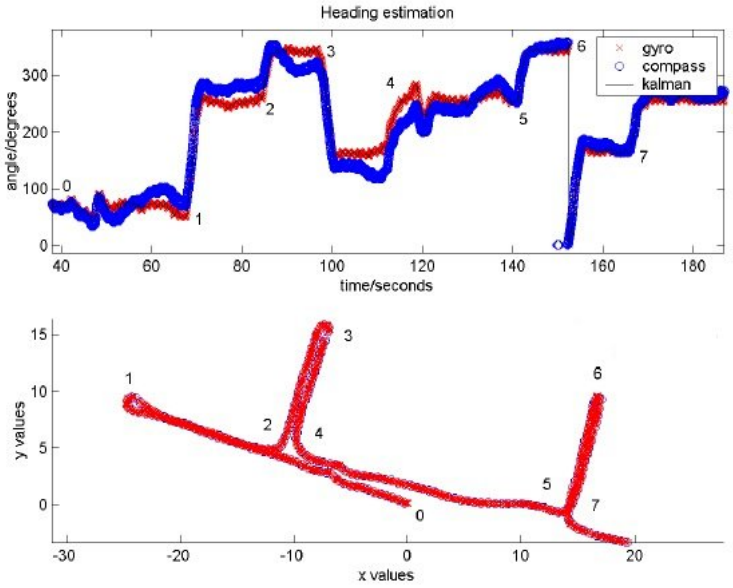


Figure 4.3: Heading Estimation. The upper image shows the gyro, compass measurement, and estimated heading. The lower image shows the dependency of the measurement on the location (indicated by numbers in both images).

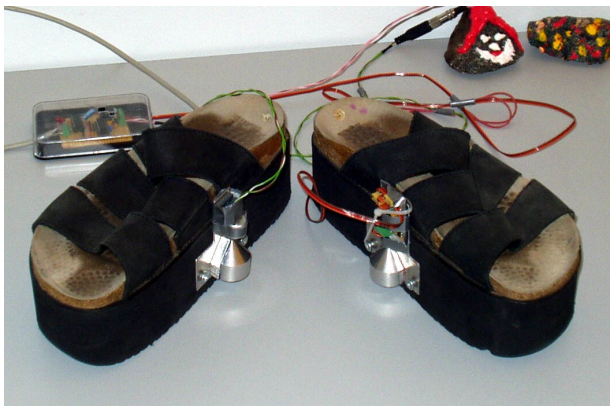


Figure 4.4: The Stride Length Measurement Unit

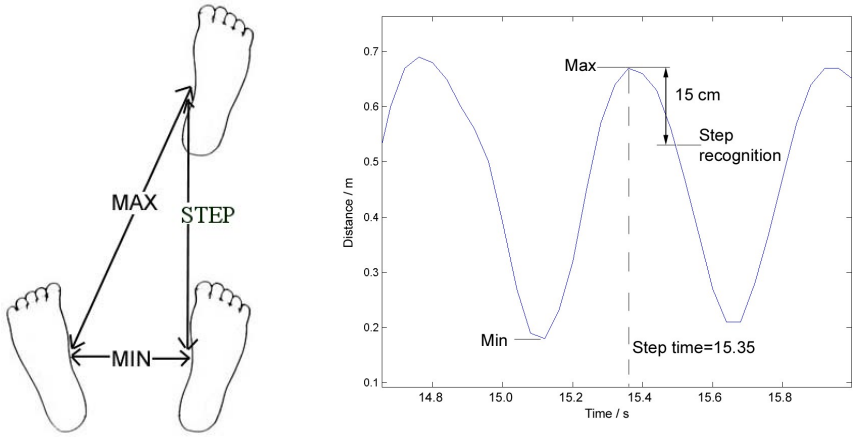


Figure 4.5: Step calculation principle

The complete steps are recognised by detecting the minima and the maxima of the pattern. The step detection is based on the detection of maxima from the signal. The step is considered to be completed when the distance is at least 15 cm smaller than the last maximum. The last maximum and the minimum are stored and used in the calculation of the step length. The time when the last maximum was found is also available and can be used as a time stamp for the step(see Figure 4.5).

As illustrated in Figure 4.5, a step forms a right-angled triangle. The minimum leg is measured when the foot passes the other foot. The maximum distance from foot to foot is the hypotenuse. After a complete step, the step length l is calculated by using Equation 4.1.

$$l_{step} = \sqrt{d_{max}^2 - d_{min}^2}, \tag{4.1}$$

where d_{max} is the maximum and d_{min} the minimum measured distance in one gait cycle (see Figure 4.5).

Another approach to the estimation of step length is the Non-Ultrasonic Pedestrian Pedometer Unit(NUPPU)[2]. NUPPU is an accelerometer-based substitute for SiLMU for situations in which ultrasound cannot be used (e.g. because of interference with other ultrasound devices). The system consists of a control unit and two 2D accelerometer units (one in each foot). The control unit reads the analogue signals from the accelerometers, makes a 10-bit A/D conversion, and writes the data to the serial port of a PC at the sampling frequency of 100 Hz. The accelerometers have a measuring range of $\pm 1.5g$ and there are two accelerometers perpendicular to each other in one accelerometer unit.

The accelerometers were placed on the feet to measure the forward acceleration and the sideways acceleration (see Figure 4.6). In this configuration it is possible

[2]one needs to be a Finn to understand the play with words. Basically both abbreviations mean bud in Finnish.

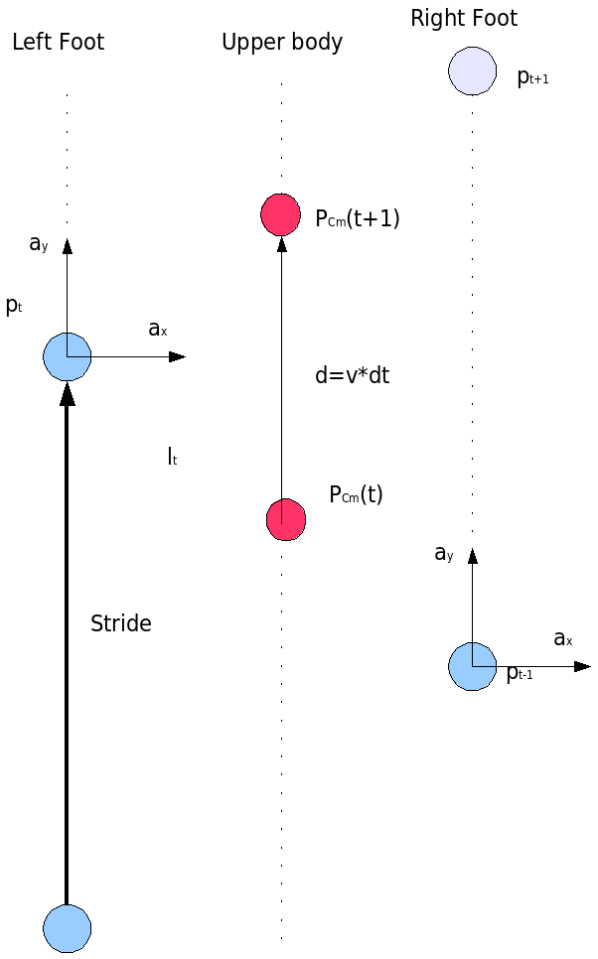


Figure 4.6: Accelerometer-based step length estimation

to detect the direction of a step, but the direct estimation of the step or the stride length is not possible. Figure 3.13 shows that the leg is in its correct orientation only during the mid-stance phase. When starting the initial swing the leg orientation is pitched almost 90 degrees towards the ground plane, which effectively cancels out all effects on the accelerometer, which is in the configuration shown in Figure 4.6. Because of this, the tracking of the foot over the whole gait cycle is impossible. Instead, as an indication of the leg speed, the deceleration phase of the leg is used. After the initial contact the leg is almost in the correct plane for this accelerometer configuration. Thus, measuring the deceleration phase and integrating it into the speed information will give an indication of how fast the leg was moving.

Practically, the estimation is based on the detection of stance phases. The reconstruction of the speed information is obtained only after the leg has reached the stance phase. The continuous measurement is obtained by updating the speed estimate of the upper body (and the distance travelled by the upper body is obtained by measuring the time from the previous step).

4.3.3 Upper body location estimation

The laser dead reckoning requires continuous estimation of the movement of the upper body. The upper body is estimated to be between the legs at all times. The estimation process is illustrated in Figure 4.7. The figure illustrates "continuous" walking, so that the time index grows from the bottom to the top and the right/left foot dots represent stance phases. The distance the upper body (or centre of mass) has travelled can be approximated as being half of the stride distance the foot travels.

At time t the left leg has reached the stance phase and the right leg starts its swing. The distance between the legs is at its maximum ($l_{meas(t)}$ in Figure 4.7) and the step length is calculated with Equation 4.1 to be l_t. The position of the upper body is updated as $p_{cm}(t)$, which is used as a reference. The distance that the upper body has traveled in time $t + dt$ can be approximated as:

$$d_{t+dt} = \begin{cases} d_0 - \frac{1}{2}\sqrt{l^2_{meas(t+dt)} - min^2_{t-1}} & , if\ new\ minimum\ is\ not\ detected \\ d_0 + \frac{1}{2}\sqrt{l^2_{meas(t+dt)} - min^2_{t}} & , else \end{cases}, \quad (4.2)$$

where $d_0 = \frac{1}{2}\sqrt{l^2_{meas(t)} - min^2_{t-1}}$ is the approximate distance to the centreline calculated when the last step was taken. The minimum (indicated by the variable min) distance between the feet is updated every time a new minimum is detected. The final distance that the upper body moves within the whole step equals the step length given by Equation 4.1.

The reference point changes when a new complete step is recognised. The position estimate is calculated by using the heading and the obtained step length measurements. The heading of the upper body is assumed to be fixed when a human has both legs on the floor. If a new complete step has been taken, the position is updated permanently using Equation 4.3, otherwise, only a temporary estimation for the position is calculated.

$$p_{cm}(t+1) = p_{cm}(t) + \begin{bmatrix} l_{t+1}cos(\varphi) \\ l_{t+1}sin(\varphi) \end{bmatrix}, \quad (4.3)$$

where φ is the absolute heading direction in which the step was taken.

4.3.4 Laser-Based Dead Reckoning

Laser-based dead reckoning has been successfully used in robotics applications with promising results. However, there are a few differences when the methods are being used for personal dead reckoning. In mobile robots the sensor is usually installed statically on top of the robot, which is not possible in the case of a human. Walking (or in the worst case, running) is a dynamic process, in which the whole body is

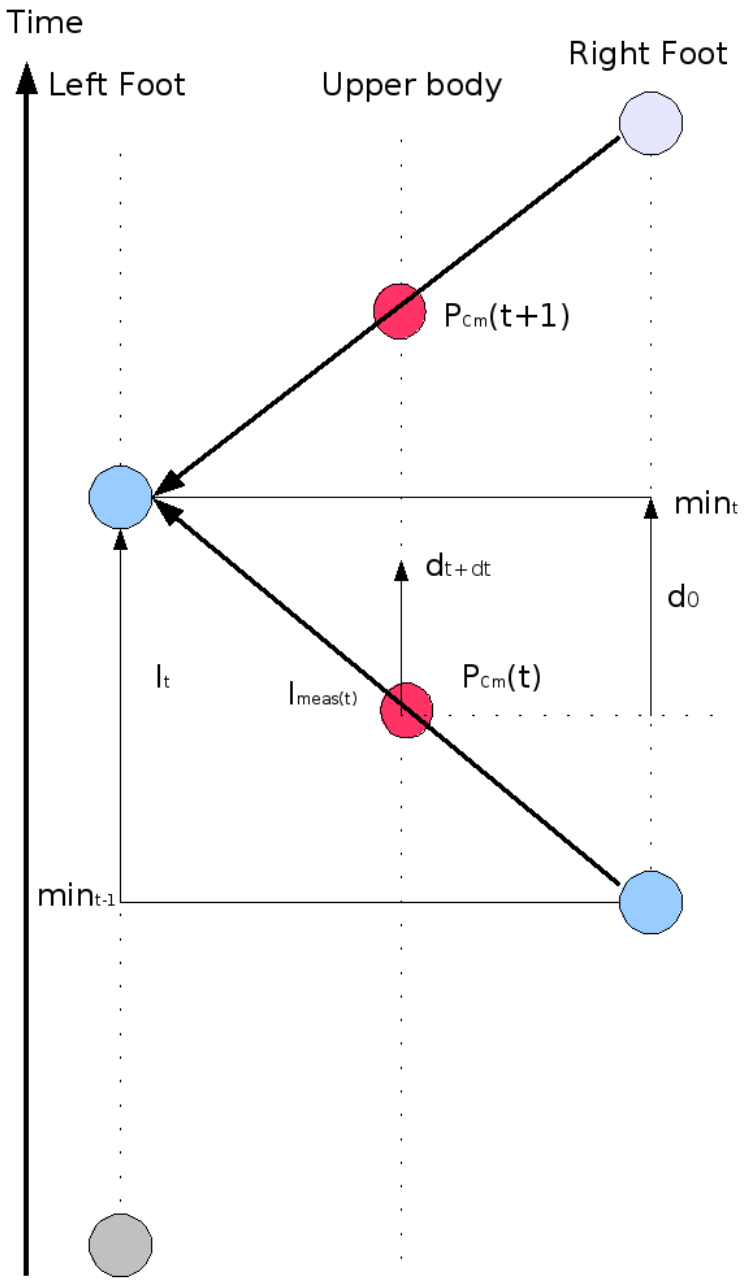


Figure 4.7: Upper body movement estimation

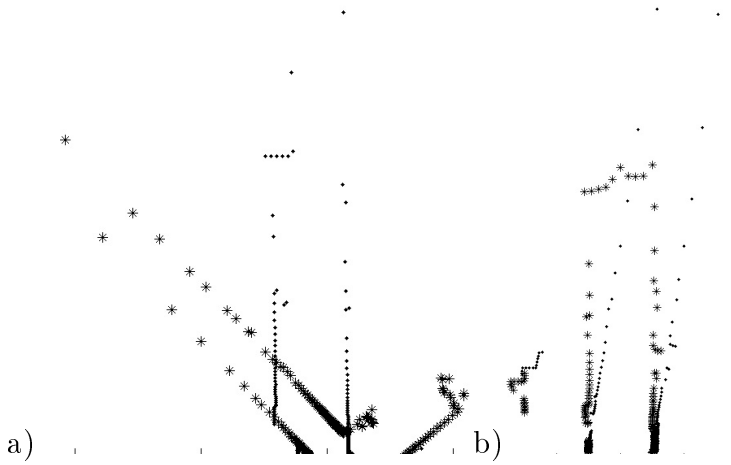


Figure 4.8: Problems of scan matching. a) The heading error can be significant between scans. This also causes many data points to be outliers. b) The two consecutive scans. The current measurement is influenced by the floor reflections.

involved. During the movement the body rotates around all three axes. The rotation around the direction in which the person is heading (yaw) causes large differences between consecutive scans (Figure 4.8a). The heading variations are typically from 20 to 40 degrees/s, but the worst case can be up to 200 degrees/s (when turning rapidly). The rotations around the pitch and the roll axes cause the violation of the 2D assumption. The changes in the roll cause e.g. the apparent corridor width not to be static.

The most problematic aspect is the variations in pitch. When the laser rotates around the pitch axis it starts to measure the floor or the ceiling (Figure 4.8b). In corridor-type environments this causes problems for algorithms that rely on geometric features such as lines. This is also a problem for the mapping. The map cannot be constructed directly from line-segmented scans, since a floor reflection would appear on the map as an obstacle.

In environments constructed by humans the vertical installations are very dominant. If the pitch angle α differs from the horizontal plane, the measured distance $\hat{L}$ can be calculated from Equation 4.4:

$$\hat{l} = \frac{l}{cos(\alpha)} \tag{4.4}$$

where L is the real distance to the object. If the variations in α are within (0...10 degrees) the variations in distance are $(0 \ldots 1{,}5\%)$, which means that the bias in the observed distance to the object is a less significant error source than the floor (and ceiling) reflections. If the sensor is mounted on the body at a height of 1.2 m and is tilted 10 degrees, the sensor is measuring the floor from a distance of 6.8 m.

Another difference to the robotic application is that the motion control inputs for the system are unknown. This makes the prediction of the movement impossible.

The normal speed of a human is around 1 m/s and the motion is usually forward. However, the speed can vary and the possibility of movements backwards and sideways should also be considered. If the laser dead reckoning is calculated five times per second, the search volume is $(\pm 0.4m, \pm 0.4m, \pm 40degrees)$, if a maximum speed of $2m/s$ and $200deg/s$ is assumed. The dead reckoning output can be used to reduce the search volume. However, it is important to have a realistic estimate of the error that the given estimate has. An average error of a few per cent of the distance travelled does not mean that the error is always within a few per cent. The error at any given moment may be tens of per cent points and for laser dead reckoning the worst case estimate should be used (otherwise the real pose can be outside the search volume).

4.3.4.1 Correlation in pose space

The large search space and biases in the measurements effectively rule out many of the scan-matching methods used in the literature. Correlation-based methods have the advantage of finding the maximum overlap between two inputs, regardless of the type of inputs. Because of this, a simple correlation-based algorithm was implemented to test the feasibility of scan matching in personal navigation. The proposed algorithm is a variant of the algorithm presented by Selkäinaho [129]. In its original form the algorithm was used for the outdoor localisation of a service robot. The algorithm makes a uniform pose distribution, which is searched through to find the pose that maximises the correlation function. The search space was limited by the robot kinematics and odometry. The algorithm is similar to the one presented by Schultz and Adams [126], except that the matching is performed with raw data instead of evidence grids.

Algorithm 4 presents the method in its basic form. The algorithm uses a given discretation values to generate the pose space and the correlation is calculated for every possible pose. As an output the algorithm calculates the relative movement from the reference scan to the current scan, as described in Section 3.3.2. The method is a brute force method in the sense that the pose search does not use any search algorithms (e.g. similar to [126]). However, the algorithm is a good tool to analyse whether the correlation works for the scan matching in this case. There are a couple of tricks to ease the computational load. The reference scan is sorted (in Line 1) and the correlation function uses a fixed threshold for checking if the two points match. This simplifies the nearest neighbour search, because the comparison can be made first in the coordinates that were sorted (e.g. only comparing x-values) and the distance has to be computed only to those points which have a coordinate closer than the threshold.

The computational complexity of the algorithm is $O(N_x N_y N_\alpha N log(M))$, where N_x is the number of poses to search in the x-direction, N_y in the y-direction, N_α is the number of headings, N is the number of the points in the fitted scan and M is the number of the points in the reference scan. Without initial guess and using 5-cm x 5-cm x 0.5-deg resolution, the algorithm requires over forty thousand pose to be

Algorithm 4 Correlation in the pose space

Inputs:

- Initial movement with respect to the reference scan: $p_i = (dx_i, dy_i, d\varphi_i)$
- Reference scan s_{t-1} and the current measurement s_t

1. Transform s_{t-1} to Cartesian coordinates m_{t-1} according to Equation 3.34 and sort e.g. according to the x-coordinates
2. Generate a set of poses, relative to the reference scan (even distribution, initial position in the middle of the grid).
3. For each pose p_n
   (a) transform s_t according to p_n using Equation 3.35.
   (b) calculate correlation score:

$$Cor(p_n) = \sum_{i=1}^{N} \begin{cases} 0, & d_{nn}(i) > t_h \\ 1, & d_{nn}(i) \leq t_h \end{cases} , \tag{4.5}$$

   where N is the number of points (measurements) in one scan, t_h is the distance threshold and, $d_{nn}(i)$ is the nearest neighborhood distance function:

$$d_{nn}(i) = \underset{j}{argmin}(\sqrt{(x_i - x_j)^2 + (y_i - y_j)^2}, \tag{4.6}$$

   (c) Store the largest correlation score and respective pose value
4. Return the pose with the best correlation score.

computed.

The most significant source of the computational load is the heading. Using the initial estimate from the PDR module reduces the search volume significantly. The heading estimate reduces the required search from 80 deg to approximately 4-8 deg (since the synchronisation is not perfect). The relative movement provided by SiLMU provides an estimate which is, on average, 5% of the distance walked, but can vary in the short term. Finally, a search volume 60cm x 36cm x ±4 degrees was used[3]. The algorithm runs in real time using a 1-GHz Pentium when using 6-cm x 6-cm and 0.6 degrees with a 6-cm threshold.

A few modifications were used in Algorithm 4 to improve the result:

1. coarse-to-fine search
2. holding reference scan over multiple scans
3. calculate the least squares estimate for the selected pose.

The coarse-to-fine search was implemented to obtain a recursive pose search. The idea is simply to start with a larger search volume with lower resolution and use the result with a smaller search volume and with higher resolution to gain greater accuracy. This process can reduce the number of pose calculations required. The problem is that the heading resolution cannot be increased (at least not much), because the heading difference between scans causes distant points to deviate from each other. For example, consider the case where the same point is measured in both scans and the distance from the origin of the laser to the points is 5 m. Having a 2-deg heading error between the scans, causes the distance between the points to be 17 cm (according to Equation 4.6). Because of this the coarse-to-fine search may give biased results.

In practice Algorithm 4 produces a static error (the magnitude is about half of the given resolution) every time the match is made. In many cases the reference scan holds enough information for matching between many successive future scans. The benefit of not changing the reference scan every time is that the error given by the algorithm is relative to the reference scan. Therefore, the error is not accumulated when scans are matched against the static reference scan. The decision on changing the reference scan is made according to the number of matching point pairs between the scans. In our experiments 200 (out of 361) hits were used.

Finally, the algorithm can output the point pairs found for the matching pose. It can be assumed that the pairs are true pairs, given that the number of point pairs is high. In this case the pose can be fine-tuned using Algorithm 3 to obtain the least squares correction. If the number of matches found is not high enough, then the algorithm returns the pose calculated by correlation (in this case Algorithm 3 may provide a worse estimate).

[3]The search area for heading was a function of initial guess. The larger the initial guess was, the larger the search area.

Algorithm 5 Overview of the combined angle and position correlation scan matching

- Initial movement with respect to the reference scan: $p_i = (dx_i, dy_i, d\varphi_i)$
- Reference scan s_{t-1} and the current measurement s_t

1. Transform s_{t-1} to Cartesian coordinates m_{t-1} according to Equation 3.34.
2. Transform s_t according to p_i by using Equation 3.35.
3. Calculate the heading difference $d\varphi_{hist}$ (or set of candidates) by using angle histogram correlation (Algorithm 6).
4. Using $\hat{p}_i = (dx_i, dy_i, d\varphi_{hist})$ transform s_t by using Equation 3.35.
5. Compute the translation (dx_{corr}, dy_{corr}) with maximum correlation by using the position correlation (Algorithm 7).
6. Return $p_{out} = (dx_{corr}, dy_{corr}, d\varphi_{hist})$.

4.3.4.2 Combined angle histogram correlation and 2D position grid correlation

The method presented in the previous subsection is good for testing the feasibility of the scan matching. It provides sufficient accuracy in real time and it works in practically all conditions (outdoors, polygonal...). However, it is not suitable if more accuracy is needed with less computational power (e.g. for map-based localisation). The latter was of more concern when investigation aimed at finding more efficient solutions was started. In the early phase of the work the histogram matching presented in [93] (based on [143]) was tested. The test showed that the heading correction was robust, while the translational shift was more error-sensitive. Having implemented Algorithm 4, the idea for the next step was to estimate the heading separately and then use correlation to find the translational shift.

The advantage is that instead of $O(N^3)$ calculations, now only $O(N + N^2)$ calculations are made. Moreover, having a good heading estimate helps the correlation, because the two scans are properly aligned, and thus the distance between points is the true distance to the nearest neighbour, which in turn allows a recursive search in the position space.

The steps of the algorithm are illustrated in Figure 4.9 and explained in Algorithm 5. Algorithm 5 is an overview which explains the whole process. The steps of the angle histogram correlation are presented in Algorithm 6 and the steps of the position correlation are presented in Algorithm 7.

The angle between two laser scans is corrected by using the angle histogram of the scan [143]. Additionally, two other alternatives were tested, but the histogram calculation based on the raw laser measurements was found to be the best [65]. An

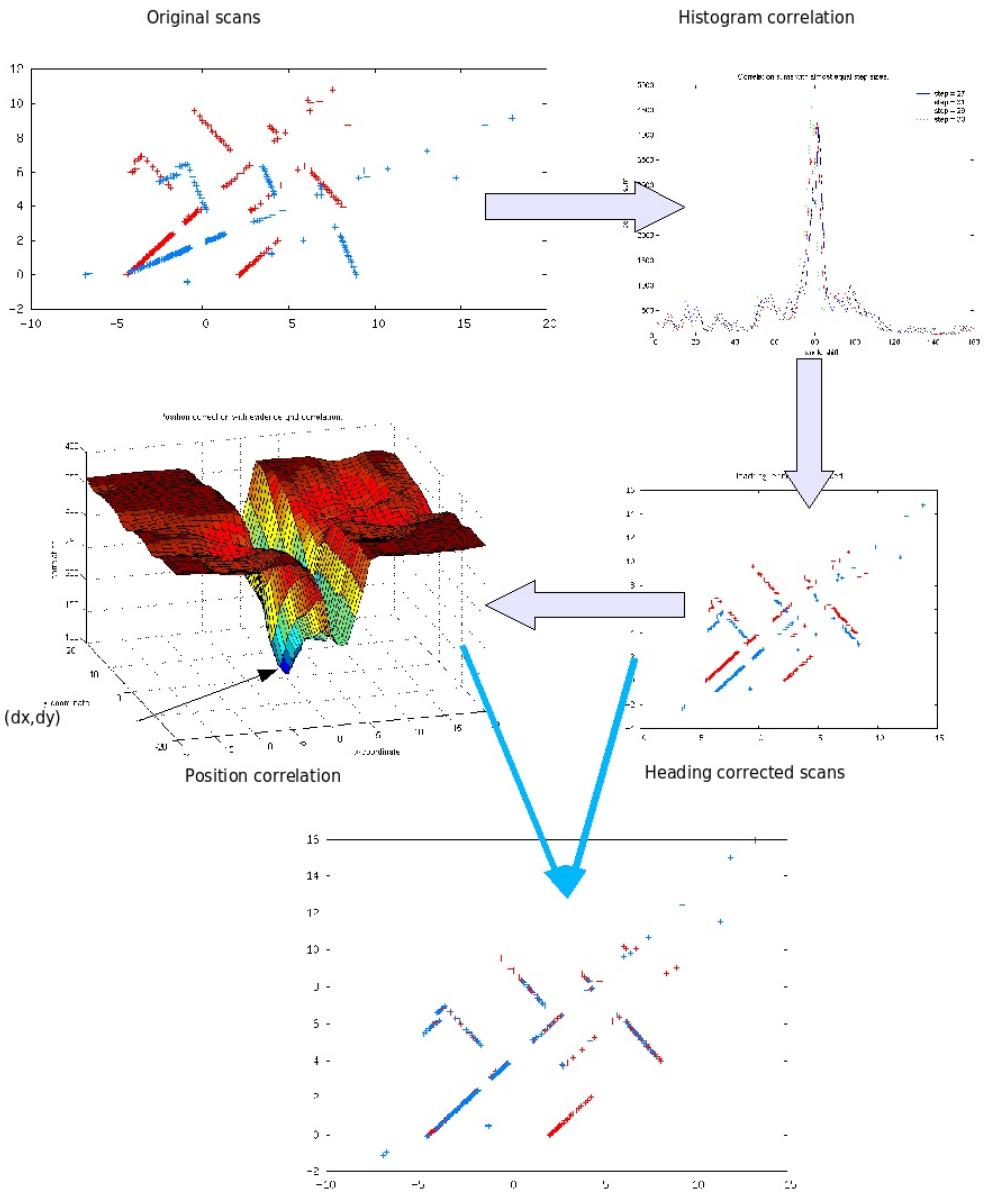


Figure 4.9: Overview of combined angle and position correlation scan matching

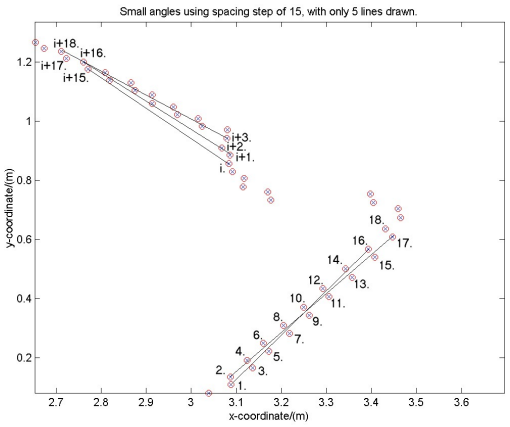


Figure 4.10: Laser scans are extracted into angles for the histogram correlation.

angle histogram represents the frequency of the discretised tangent values (see Figure 4.10) of the polygonal objects in the environment. Because the laser scan is ordered, the tangent values can be approximated by calculating the angle using N consecutive points in the scan. The N points are required to reduce the measurement noise. As an example, in Figure 4.10 the angle between Points 1 and 2 differs significantly from the angle between Points 1 and 15. The number of points used in the calculation has to be large enough to smooth the measurement inaccuracies but small enough for some details to be able to be seen. A number close to 30 was found to be the most reliable when using laser scans with 361 points.

The histogram is calculated for both scans by discretising the angle space and by calculating the frequency of each angle value. The angle shift (correction) is directly obtained from the shift that maximises the cross-correlation (Equation 3.40) of the histograms being used.

The complete method is presented in Algorithm 6. The algorithm does not explicitly extract lines from the scans, but naturally it performs best in polygonal environments. The most accurate estimates are obtained when there are long lines in the scans, e.g. walls.

The algorithm can be made more robust by using different step sizes in the angle estimation. The result is then a set of different angle candidates from which the translation correction can be calculated. The candidates can be used to give an approximation of the estimation error. For example, in Figure 4.11 the maximum of the correlation differs slightly between the different step sizes.

The translation correction returns the best correlating shift in the x- and y-directions between two scans. The correction is calculated only for the translation and it is assumed that the rotation between the scans is already corrected. The complete algorithm is shown in Algorithm 7.

An example of the correlation results is shown in Figure 4.12. The figure shows the correlation values of each point in the search grid (xy-plane). A low value means a good correlation. Figure 4.12 represents a typical corridor; there is a good match in

Algorithm 6 Angle Histogram Correlation

- Reference scan s_{t-1} and current measurement s_t

1. Transform s_{t-1} and s_t to Cartesian coordinates according to Equation 3.34
2. Calculate angle values from both laser scans: $\alpha_i = atan(\frac{y_i - y_{i-N}}{x_i - x_{i-N}})$, where N is a constant number of points skipped in the scan.
3. Discretise the calculated angles and calculate the frequency count i.e. the angle histogram
4. Calculate correlation (S) for the angle histograms (A and B) over the desired search area.

$$h(k) = \sum_{i=0}^{N} h'(i) \cdot h(i+k), \tag{4.7}$$

   where N is the number of possible discrete values in the histogram.
5. Save all local maximum correlations and the corresponding angles.
6. Sort these correlations and return the desired number of best correlations and angles.

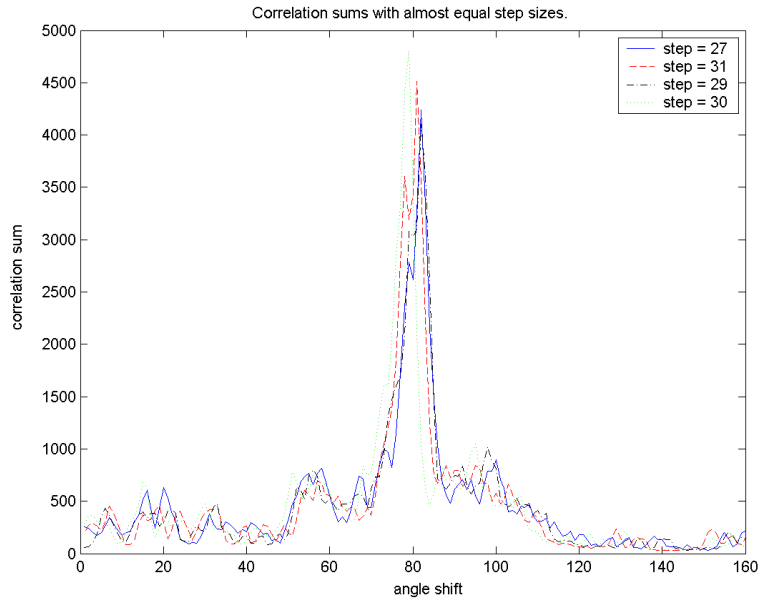


Figure 4.11: Angle cross-correlation with slightly different angle step sizes.

Algorithm 7 Translation correction using correlation (2D)

Inputs:

- Reference scan s_{t-1} and current scan s_t
- Initial movement with respect to the reference scan: $p_i = (dx_i, dy_i, d\varphi_i)$, where $d\varphi_i$ should be the corrected heading difference between the scans

1. Transform s_{t-1} to Cartesian coordinates m_{t-1} according to Equation 3.34
2. Transform s_t to m_t and transform according to p_i by using Equation 3.35.
3. The reference scan coordinates m_{t-1} are discretised e.g. $x \in [-10, 10]$ and $y \in [0, 15]$ to a set of new points m^d_{t-1}
4. Coordinates in m^d_{t-1} are sorted according to x- or y-coordinate values using the Quicksort algorithm [25].
5. Create a position search grid based on the desired search area and resolution.
6. For each point p_k in the position grid
   (a) Translate m_t according to p_k (simple summation, because the rotation is fixed)
      i. For every point in the current scan, calculate the distance to the nearest neighbour in the reference scan by using Equation 4.6.
      ii. Calculate the correlation value for the position p_k

$$Cor(p_k) = \sum_{i=1}^{N} \begin{cases} \frac{d_{nn}(i)}{D} & , d_{nn}(i) <= D \\ 1 & , d_{nn}(i) > D \end{cases} \tag{4.8}$$

      where $d_{nn}(i)$ is the distance of i:th point in the current scan to the nearest neighbor in the reference scan (according to Equation 4.6) and D is the maximum distance allowed for the nearest neighbour (larger values indicate outliers).

7. Return $p_{out} = (dx_i + p_x^{min}, dy_i + p_y^{min}, d\varphi_{in})$, where (p_x^{min}, p_y^{min}) is the translation that has the smallest correlation value.

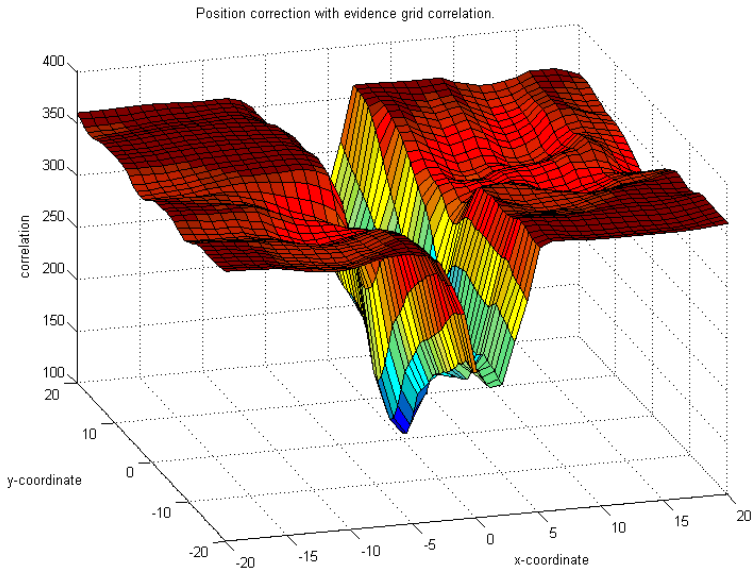


Figure 4.12: Correlation values for one search grid. Small values indicate better match.

one direction because of the corridor walls, but in the other direction the correlation forms a valley of uncertainty, with a less visible minimum.

The minimum distance to the nearest neighbour is a more consistent measure of correlation than in the previous method (section 4.3.4.1), in which the heading also affected the correlation function. Because of this, it is possible to use a similar iterated searcher to that presented in [126]. The searcher divides the initial search space (a 2D grid in this case) into 16 cells, placing the initial estimate in the middle. In the next iteration, the cell with the best correlation is taken as the centre and the search area is divided into half and the resolution doubled. This process is repeated until the required resolution is reached. The process is illustrated in Figure 4.13. The red area represents the best estimate found from the previous iteration and the dashed (green) area represents the search area for the next iteration. The areas between two consecutive searches overlap, which is required to guarantee a correct solution for the case where the true position is found from the border of the best correlating cell.

The iterative searcher reduces the required computation significantly. Consider an example of searching an area $(\pm 1.28m, \pm 1.28m)$ with a resolution of $0.02m$. Using uniform distribution as the search grid, the required number of positions to be calculated is $128 \times 128 = 16384$. Using the iterative approach the required amount of iterations is $N = log_2(\frac{1.28}{0.02}) = 6$, which gives a total of $16 \times 6 = 96$ positions.

Comparing the method to the one presented in Algorithm 4, this method is much faster, but it assumes that the heading is correctly corrected with the angle histogram method, which in turn assumes that the environment is polygonal (or polygonal enough). One way to increase the robustness is to use a set of heading corrections obtained from the angle histogram algorithm. As illustrated in Figure 4.11, the use

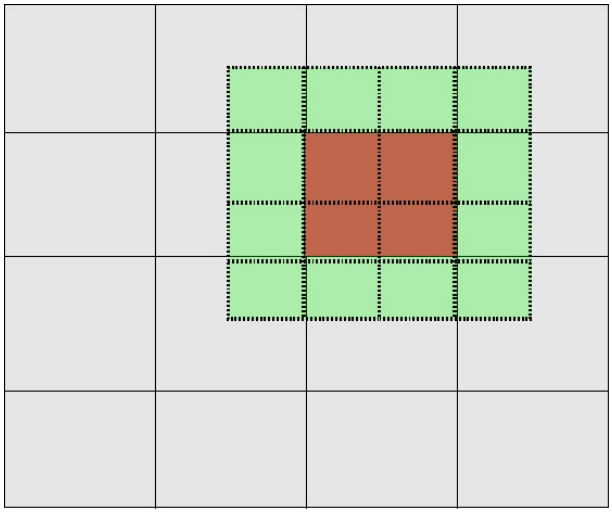


Figure 4.13: Iterative position searcher

of a different number of points in the calculation of the tangent angles produces slightly different results. Algorithm 7 can be run with different fixed heading values to find the maximum correlation between these.

4.3.5 Increasing Error tolerance

Laser dead reckoning may fail if: 1) the real movement between the reference scan and the current measurement is not within the search space; 2) there is not enough information in the scans to provide a unique result; 3) the dynamics violate the assumption that the scans "represent the same environment", and 4) a combination of the first three error sources. The methods presented earlier will fail if the search area is not sufficiently large. Increasing the search area solves the problem, but opens up the possibility of other error sources. The second point is always a problem for scan matching. For example, long corridors in an office environment provide very little information in the direction of the corridor, which in turn can cause a situation where there are several locations that give approximately the same correlation. A dynamic environment is also usually a problem for localisation systems. For example, a large amount of people (usually present in the case of a demonstration) moving in front of the laser causes apparent movement between the scans, even when the laser has not moved at all. Opening a door was found to be one of the most challenging cases for laser-based dead reckoning. When the door opens, the movement is interpreted as rotation, even when the laser is stationary. The problem is illustrated in Figure 4.14; after the door is opened the heading error is 45 degrees.

The static reference scan approach was implemented to reduce the error of scan matching (Figure 4.15). This, however, can cause a combined error effect in some situations. Consider the situation in Figure 4.15b. The reference is changed at time t. This reference is used in future time steps $t+1,...t+N$, until the decision is made to change the reference. The estimate from the previous matching was

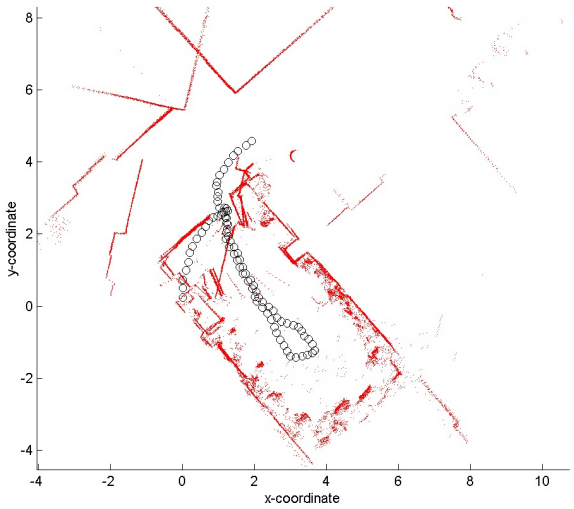


Figure 4.14: An opening door causes severe problems if only the laser is used for matching.

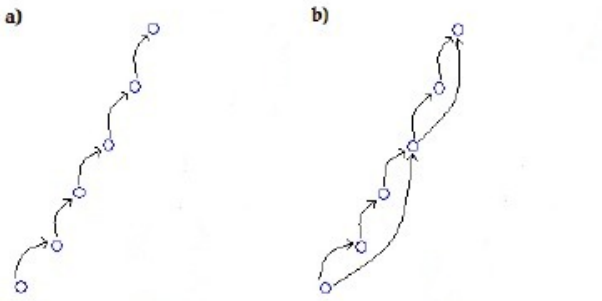


Figure 4.15: Two alternative ways to choose the reference scan. a) changing the reference every time a new scan is received; b) changing the reference only when necessary. Courtesy of Seppo Heikkilä [65].

used as the initial estimate of movement. Now, if the subject is walking through a repetitive corridor, it is possible that at some time step the local minimum will be found instead of the correct displacement between the reference scan and the current measurement. Using that result as the initial guess for the following rounds can cause the true displacement not to be within the search area.

The same process can also cause more sensitivity to dynamics: a sudden change in the view is more likely to cause error if there is less correlation between the reference and the current measurement.

There is no unified solution to all situations. In principle, having a good initial estimate of displacement helps, because then the search area can be kept small, which reduces the possibility of local minima and reduces the possible maximum error. However, as mentioned earlier the search area must take into consideration the

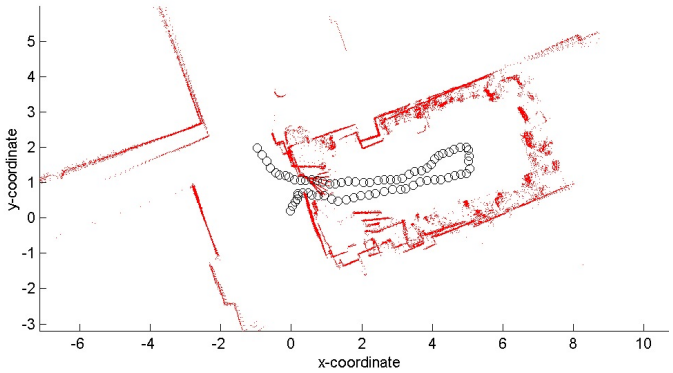


Figure 4.16: Result of opening door with error checking

worst-case error of the dead reckoning (otherwise the method is tolerant to the case in which the true pose is outside the search area). In this case the current pose estimate with respect to the reference scan was calculated: 1) using dead reckoning only; 2) by calculating the displacement between every scan using laser dead reckoning (Figure 4.15a) and 3) calculating the displacement between the reference and the current measurement (Figure 4.15b). In all three cases the result should be the same. If not, then a rule-based method was used to find out which estimate to use. (If the error is only visible in the third case, the pose from the previous round can be used and the reference set accordingly; if both matchers fail, then the initial estimate is the best one to use). Figure 4.16 shows the result of this process.

4.4 Map-Based Localisation

The benefit of dead reckoning is that it is fully infrastructure-free. However, the integration of the sequential movements eventually increases the error cumulatively to a significant level and the location is lost. One way to reduce this error is to use map-based localisation methods. The framework for continuous probabilistic localisation was presented in Section 3.1.2. This section presents an implementation of three Monte Carlo Localisation (MCL) methods for map-based localisation:

- topological MCL
- range scan-based MCL
- combination of the above two methods

The methods use dead reckoning and maps as inputs. The methods can use any dead reckoning input (SiLMU-based, laser-based, wheel odometry, inertial navigation) and as such these methods can be used in combination with any dead reckoning

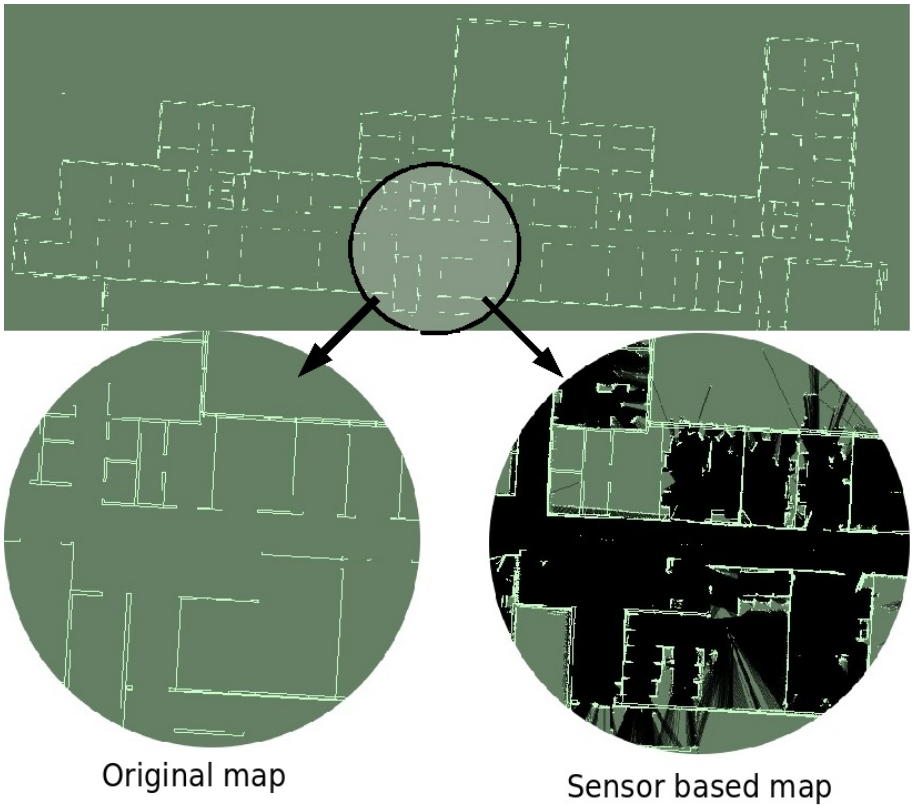


Figure 4.17: Geometric map representation versus the sensor-based map

system. What is more important is that the error of the dead reckoning estimate has a zero mean (i.e. the measurement is not biased) and that the variance is known.

The map has a significant meaning in the process. The likelihood of a pose is calculated by finding how likely the measurement is, given the current pose. The likelihood calculation requires the measurement to be predicted against the pose and map being tested. Perfect prediction requires the map to represent the actual state of the environment as the sensor measures it. However, a static map can represent only a single state of the environment. The situation is illustrated in Figure 4.17. The overall map in the figure is derived from the CAD drawing of the building. The model represents the state of the building (relatively) accurately, just at the time the building was built. Since then the rooms have been filled with furniture, cupboards, and, in some cases, with extra walls. The zoom-in pictures in Figure 4.17 show the time instant at which the tests were made. The sensor measures a radically different state of the environment than what would be expected according to the original map.

The use of the sensor-based map presented in Figure 4.17 would provide accurate measurement predictions, but it would come at the cost of actually: 1) having the sensor-based model, and 2) maintaining the proper model over time. In this thesis it is assumed that the map is a geometric representation that represents the structural body of the building (named “Original map” in Figure 4.17). As a consequence

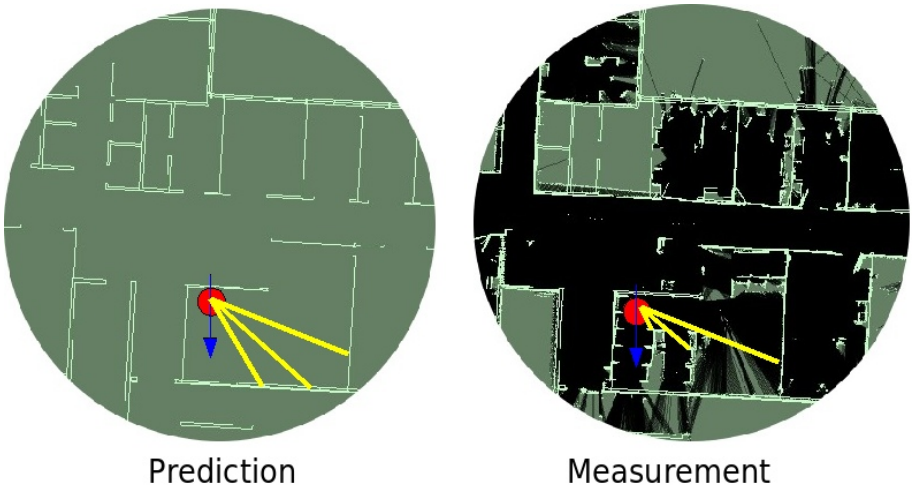


Figure 4.18: Measurement prediction versus "real" measurement using partially correct map

the methods presented should be more general (in terms of applicability) but at the cost of losing accuracy. In practice this means that the real environment is likely to have obstacles that are not mapped. The consequence is that the predicted measurement is often longer than the real measurement. Figure 4.18 shows an example. The prediction calculates the expected measurement from a given pose (red dot). According to the map the sensor would measure the distance to the walls, but in reality only the distance to the cupboards is measured. Because of this, one measurement must not have too much significance attached to the weights of the particles. This is achieved by increasing the measurement variance and by rejecting the measurements which are significantly shorter than the predicted measurements.

The initial estimate of the pose is required to limit the number of particles. In principle MCL can be used for global initialisation, but for real-time purposes the number of particles required for initialisation grows too large with large maps. Basically the initial estimate can be given in two ways: 1) the initial pose is approximated to be within some area, without further knowledge, or 2) the initial pose is approximately known. The first case is essentially the same as the global initialisation, only in a limited area. The process is illustrated in Figure 4.19. At the beginning the pose distribution is uniformly distributed over an area approximately 10 m x 10 m (and a 360-degree heading). In the first three (top) images, the distribution is spread and there is no visible solution. In the three images in the middle the number of solutions (modalities) is reduced to a few and in the bottom images the solution finally converges into one distribution. The second case initialises the pose as a single distribution, i.e. the pose is known and the particles are spread around it using normal distribution.

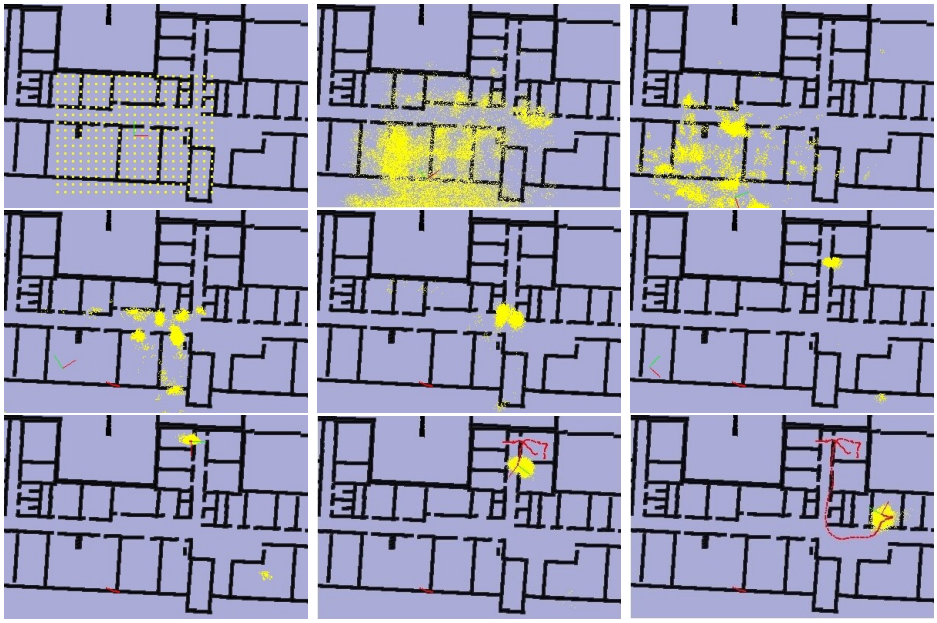


Figure 4.19: MCL initialization using uniform distribution

4.4.1 Monte Carlo Localisation Algorithms

The basic MCL algorithm is described in Algorithm 1. The algorithm describes the three steps of the MCL:

1. prediction
2. measurement update
3. sampling importance resampling

All the methods presented in this section use the same prediction and SIR steps (see Section 4.4.1.4). The prediction step takes a dead reckoning estimate and its error estimate as input (as described in Section 3.1.2.1). The predicted pose for each particle is calculated according to the dead reckoning input, with added noise. The zero mean normal distributed noise is added according to the error estimate.

The update step calculates the probability of each particle on the basis of the most recent measurement. This step differs between the different MCL implementations.

4.4.1.1 Range scan-based MCL

Range scan-based MCL is a traditional way to update the measurement likelihood. The range scan is a set of range measurements taken at a certain time. The source of the range measurements (sensor) is irrelevant as long as the measurement accuracy is known. This implementation closely follows what is presented in the literature (namely [53, 139]).

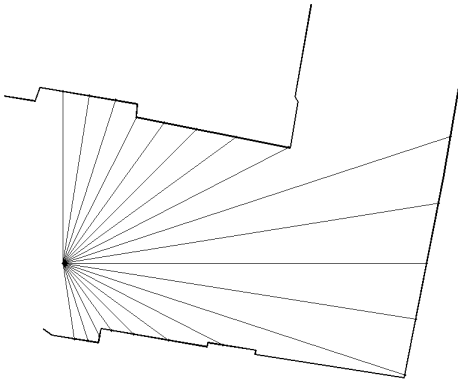


Figure 4.20: An example of a virtual scan

In this case the measurement is a laser range scan taken from a single pose in the environment. The update phase compares the expected measurement to the real measurement and calculates the likelihood of the measurement being taken from the place the particle represents. The expected measurement is calculated for each particle using the a priori map. Figure 4.20 shows an example of a virtual scan of the expected measurement calculated using a line map. The virtual scan is created by calculating the expected measurement lines and by calculating the intersections with the map lines. Similarly, the virtual scan can be calculated from the occupancy grid map using the ray tracing technique.

The likelihood calculation is the most critical part of the MCL algorithm. The weight of a single range measurement can be calculated using Equation 3.17 and if the ranges in the scan are expected to be independent then Equation 3.18 can be used to obtain the weight of the particle. A laser is an accurate measurement device (the standard deviation of measurement is less than 2 cm). Figure 4.21a illustrates the effect of measurement variance on weighting using Equation 3.17. The red curve shows the function for a laser range finder with a perfect map. Even if there were a perfect map, the problem of using such a function is that it causes very high peaks in the weights of the pose distribution. Practically all poses that are not within a few centimetres of the real state get zero weight, which in turn means that there are only a few particles that will obtain high probability. As a result the distribution will lose its capability to represent the uncertainty. Moreover, the dead reckoning always has errors that do not fit into the model, and the heading error is not properly handled by Equation 3.17.

Because the map and the environment differ in many parts, the map uncertainty is large (or biased). Because of this, one measurement (or even a few) should not affect the weights of the pose distribution too much. Figure 4.21b shows two alternative weighting functions. Both functions have a cut peak, which means that all the poses within a smaller distance than the threshold will have equal weight. The other curve additionally has a "constant level of probability", which means that the weight of a pose is not going to go into zero, no matter how bad the measurement was compared to the prediction. In a sense the constant gives a probability of bad prediction. This is the likelihood function implemented for one range measurement and for the whole

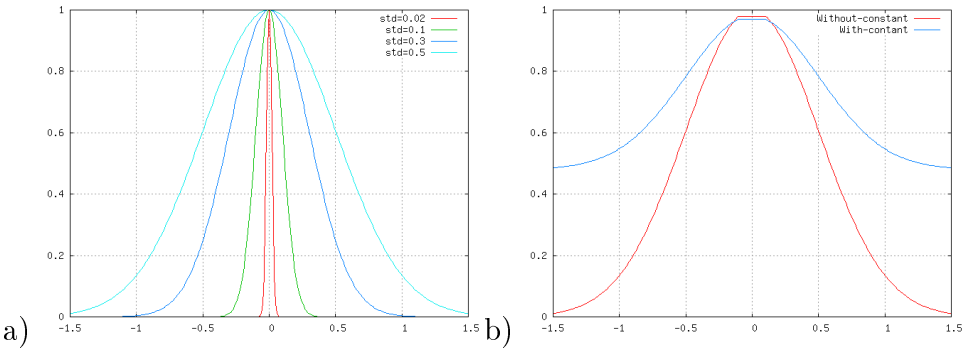


Figure 4.21: Weighting functions for a single range measurement: a) using Equation 3.17 with different weights (without scaling); b) by smoothing the peak and using the constant as "base probability".

scan the weighting of one particle can be written as:

$$p(z_t \mid x_t^i, m) \sim \prod_{j=1}^{n} (e^{\frac{-dl_j^2}{\sigma_{meas}^2}} + P_{false}), \tag{4.9}$$

where σ_{meas} is the sum of standard deviations of the map error and measurement error, dl_j is the distance difference between the predicted distance measurement and the measured value, P_{false} is a constant and n is the number of range measurements in one scan. In our implementation only ten measurements (out of 361) from the range scan were used for the update. The speed of the algorithm was the main reason for this. Furthermore, the larger amount of measurements can bias the likelihood calculation more than it helps it.

P_{false} can also be used as a function of dl: if the expected measurement is further than the measured one, P_{false} is increased. In this case it is probable that there are some unmapped dynamic objects or structures in front of the laser.

4.4.1.2 Topological MCL

Another, complementary, way of updating the weights of the particle distribution is to use the topology of the space. In [46, 141] Equation 3.33 was presented. The likelihood calculation was made for WLAN-based positioning with the rule that if the particle crosses the wall it will get zero probability. The idea can be extended to allow the updating of the likelihoods without any knowledge of the real position. Intuitively, the closer the particle is to the wall, the less likely it is. Using this intuition, an occupancy grid map can be converted into a likelihood function by using the distance transform [48]. The distance transform calculates the shortest distance to an obstacle for each cell. Thus, the outcome is a sampled 2D function, which gives the distance to the closest obstacle for each location in the map. This distance is thresholded; that is, all distances above the threshold will result in equal probability, but values closer to the wall will have a smaller probability.

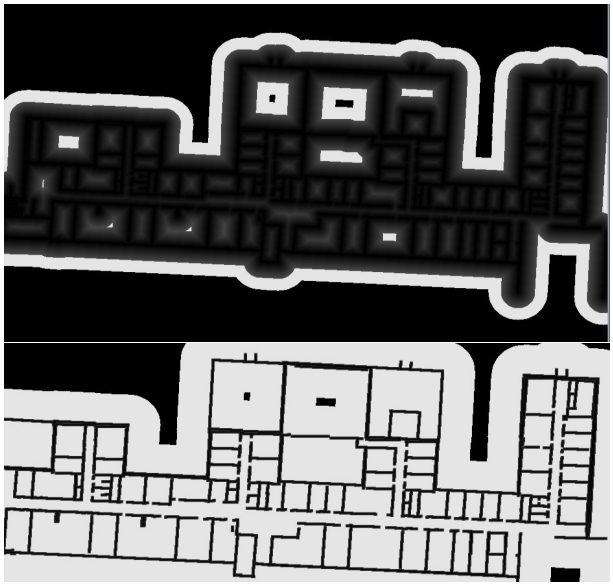


Figure 4.22: An example of a distance transform (upper) and a likelihood function (below). In the likelihood function all grey areas have equal probability and only the vicinity of the walls has low probability.

The updating of a particle's weight is directly obtained from the precomputed grid, which makes the algorithm very fast.

An example of the distance transform and precalculated weighting function is presented in Figure 4.22. The grey area in the bottom image has constant probability and the darker areas have lower probability. The "measurement" update function is given by Equation 4.10.

$$p(z_t \mid x_t^i, m) \sim \begin{cases} d(x_t^i), & if\ d(x_t^i) \leqq d_{max} \\ d_{max}, & if\ d(x_t^i) > d_{max} \end{cases}, \tag{4.10}$$

where $d(x_t^i)$ is the distance to the nearest obstacle from position x_t^i and d_{max} is maximum distance that is affecting the weighting. This weighting function is made considering personal navigation. The selected $d_{max} = 0.3m$, which means that all positions in the space that are further than 0.3 m from an obstacle will get the same probability. The weighting could take the most probable trajectories into consideration (e.g. if a robot is controlled in a certain way), but in this case a human might use the whole empty space.

The method uses a metric map and produces a metric pose distribution. It is called a topological one, because the pose update procedure matches the trajectory of the particles on the map. There is no direct measurement from the environment, but the particles that are travelling according to the true trajectory will have a better probability than those which are not. The process also requires the map to have a distinguishable topology. All the particles in the empty area have equal probabilities, and the accuracy depends on the width of the corridors, the amount of turns made,

and the number of rooms visited. Thus the deviation of the probability distribution in the long term is comparable to the size of the empty area defined by the map.

The algorithm is robust and works well with an office-type environment, even without any observation from the environment, as long as the trajectories match the topology of the environment (i.e. it is not suited for localising a robot or human which/who is moving in a single room, but is suited for localising an entity exploring the whole environment).

4.4.1.3 Combined topological and range scan based MCL

The two different weights/likelihoods can be fused. The weight for one particle becomes:

$$w_i(t+1) = w_i(t) L_r L_t, \tag{4.11}$$

where L_r is the likelihood obtained using the range measurement, L_t is the likelihood obtained from the topological likelihood function and $w_i(t)$ is the weight of the particle after the previous update. The combined algorithm was implemented so that in each iteration (arrival of a new pose) the weight is updated only by using Equation 4.10. The combined update is used only if the movement between the last measurement update and the current measurement has been more than the threshold. This was implemented to reduce the bias that can occur if a range scan is updated continuously in a single pose.

4.4.1.4 Sampling Importance Resampling

If the prediction and the update steps are continuous, the particles gradually diverge. The particles move according to the dead reckoning and the update phase only updates the probability, not the position of the particles. This problem is solved by re-sampling the distribution with the Sampling Importance Resampling (SIR) algorithm [57, 3]. This algorithm re-samples the distribution in such a way that the particles with high probabilities get duplicated many times and the particles with low probabilities vanish completely. The pseudo-code of the algorithm is introduced in Algorithm 8. The algorithm inputs the whole particle distribution. Each particle has a state and a weight w that represents the probability of the particle. The algorithm then runs through the particles, multiplying the high-probability particles and removing the low-probability particles.

Algorithm 8 selects the first particle randomly (the initialisation of U). If the particles have uniform weight it is $\frac{1}{N}$, which is used as a comparison when copying the particles (Lines 1a i to iv).

While the SIR algorithm tries to sample from the "real" distribution, it also always discretises the distribution. This causes a loss of information and therefore it is not wise to trigger the SIR update unless necessary. This decision is not necessarily easy.

Algorithm 8 Sampling importance resampling

Inputs : *Particles* χ_{t-1}
$Variables : U = 0, Q = 0, i = 1, j = 1$
Outputs : *Particles* χ_t
$U = uniformrandom()/N$

1. while U<1.0
   (a) if Q>U
      i. U+=$\frac{1}{N}$
      ii. $\chi_t^i = (x_{t-1}^i, \frac{1}{N})$
      iii. i++
   (b) else
      i. $Q+ = w_{t-1}^j$
      ii. j++
   (c) end
2. end

Intuitively, the distribution represents the real distribution well if all the particles have an equal probability. Conversely, if there are many low-probability particles and only a few high-probability particles, then the distribution does not effectively represent the real distribution any more. Therefore one measure for triggering the SIR update is the variance of the probabilities. The SIR update is triggered whenever the variance starts to grow.

4.4.1.5 Determining the pose

The particle filter returns as many hypotheses of the pose as there are particles, thus giving a probability distribution of the pose instead of an absolute value. Figure 4.23 gives an example. The cloud on the right is the distribution of the pose. The distribution covers the whole room where the PeNa user is. The continuous (red) line is calculated by using the weighted average of the whole distribution; that is

$$\tilde{p} = \sum_{i=1}^{N} w_i p_{i.} \tag{4.12}$$

The weighted average is not necessarily the true position, but it can be seen from Figure 4.23 that it gives a smooth path.

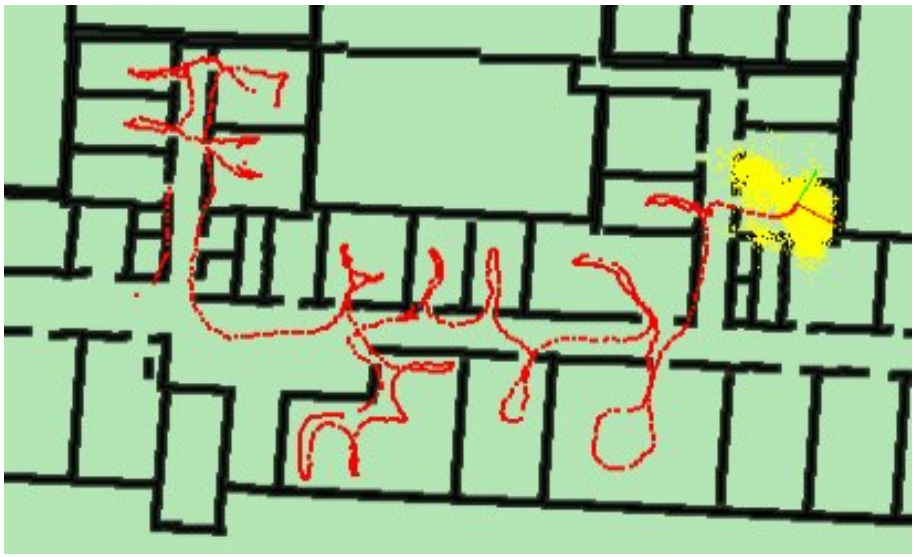


Figure 4.23: An example of a particle filter result. The dots are hypotheses of the pose at the moment. The continuous line is the trajectory returned by the weighted average of the distribution.

Chapter 5

Tests and Results

5.1 Step length tests

The accuracy of SiLMU was tested separately. The test was performed by walking along a line. In the test the measured distance is the sum of the step lengths obtained using 4.1 and the real walked distance was 55 m. The measured distance varied on both sides of the real distance, giving approximately zero mean variance. The first measurement did provide an error far bigger than the average, but no particular reason for this was found. The average error with the 55-metre distance was 0.70 metres. This gives an error percentage of about 1.28%. The maximum error was 3 metres, the error percentage being 5.45%. The complete set of results can be seen in Table 5.1.

Table 5.2 shows the results of a similar test set performed with NUPPU. The major difference between the two measurement devices is that once calibrated, SiLMU provides a reliable estimate from person to person. NUPPU, on the other hand, is based on an indirect measurement of a step length, which makes it more tolerant towards the walking style of a person. Because of this the estimate can be biased (i.e. the length is always too short or too long) when using a single calibration for different persons. This is also visible in Table 5.2. All the lengths provided by the device are reported as too short. Even if the result had been calibrated, the maximum error of the data set is more than 10% of the distance travelled (i.e more than 6 m within 60 m). Because of this, it can be stated that NUPPU provides a less accurate dead reckoning estimate, which cannot be used as such for the map-matching algorithms (mainly because the estimate can be biased). However, NUPPU was able to provide a "good enough" estimate of movement for the laser dead reckoning, even without the need for calibration between persons.

5.2 Personal Navigation tests

To test the functionality of the system, the setup was tested in an office-type environment. The test area was the 2nd floor of the TUAS building at Helsinki University

Table 5.1: Measurement results from the stand-alone test for SiLMU

Sample	Measured dist(m)	Real dist(m)	Error(m)	Error(%)
1	58.00	55	3.00	5.45
2	55.05	55	0.05	0.09
3	56.84	55	1.84	3.35
4	56.36	55	1.36	2.47
5	56.12	55	1.12	2.04
6	55.65	55	0.65	1.18
7	54.96	55	-0.04	0.07
8	54.45	55	-0.55	1.00
9	54.76	55	-0.24	0.44
10	54.88	55	-0.12	0.22
11	54.08	55	-0.92	1.67
12	54.15	55	-0.85	1.55
13	54.74	55	-0.26	0.47
14	54.67	55	-0.33	0.60
15	56.34	55	1.34	2.44
16	55.12	55	0.12	0.22
17	55.05	55	0.05	0.09
18	55.01	55	0.01	0.02
19	54.83	55	-0.17	0.31
20	53.95	55	-1.05	1.91

Table 5.2: Measurement result of NUPPU stand-alone tests

Set	Measured dist(m)	Real dist(m)	Error(m)	Error (%)
1	47.79	60	-12.21	20.34
2	58.51	60	-1.49	2.48
3	48.23	60	-11.77	19.62
4	49.68	60	-10.32	17.2
5	56.06	60	-3.94	6.57
6	55.3	60	-4.7	7.83
7	50.46	60	-9.54	15.9
8	54.27	60	-5.73	9.54
9	59.5	60	-0.5	0.83
10	59.82	60	-0.18	0.3

Figure 5.1: Example picture from testing area (above) and the map of the test area (below). The filled area is the corridor network used and the black dot marks the typical starting point of the test runs.

of Technology. The tests were performed in corridor-like environments and office rooms alongside the corridors. The test area is challenging for the laser range finder since there are plenty of class walls around (see Figure 5.1). A map of the whole test area is shown in Figure 5.1. The filled area is the corridor network and the black dot marks the start and end point of the test runs (in most of the cases).

The purpose of the tests was to demonstrate the performance of the dead reckoning system for the deliverable "D2.2 Personal Navigation System Test Report" [117]. The tests reported in this chapter use the data collected during this test. The data sets represent different natural paths that can be walked in an office. The paths include walks in corridors of different lengths and paths that include rooms and corridors. However, all the data sets are collected to represent situations in which the walker is actively exploring the environment. An illustration of all the paths is given in the appendices (Appendix B.2 illustrates the paths on a map).

The dead reckoning results for Algorithm 4 are the same as reported in [117]. The results of the map-based methods, as well as the laser dead reckoning algorithm (Algorithm 5) are post-processed. The setup had SiLMU for stride length estimation, a compass-compensated gyro, and a laser. The laser dead reckoning algorithm described in Subsection 4.3.4.1 was used. The dead reckoning results were calculated in real time while walking. The collected data include processed SiLMU pose

estimates, processed laser dead reckoning estimates, and laser measurements. The laptop computer has a 1066-MHz Pentium III Mobile CPU and the Windows XP operating system.

5.2.1 Error Estimation

The error estimation is difficult as it requires a reference. The usual error parameters for personal dead reckoning systems are the error in the distance walked or the relative error between the end and start positions after a closed loop has been walked. The first one requires a specific distance to be walked, which is only possible if the trajectory is simple. Additionally, the results may be affected by the test subject not walking in a straight line. The second measure is relatively simple to measure. The result can be obtained as long as the test subject returns to the initial position. The problem with this method is that walking a closed loop does not reveal the real errors of a dead reckoning system. The error in position depends considerably on the error in the heading, which is the most significant cause of positional displacement. Additionally, walking a closed loop does not reveal the scale; that is, the dead reckoning system may give distances that are always too short or too long.

In this case the trajectories tested are such that an exact reference would be impossible to achieve. The system was thus tested by walking a closed path. For the error analysis the end/start difference is calculated. The reference trajectory is calculated by fitting the data to the map using MCL. MCL produces a more or less topologically correct path, but it may not follow the true trajectory accurately (the derived pose is the weighted average of the probability distribution). Therefore the trajectory provided by MCL is used as a measurement to correct the calculated odometry. The fitting is formulated as a least squares minimisation problem. The estimated state is the whole trajectory:

$$x_t = \begin{bmatrix} p_1 \\ \vdots \\ p_N \end{bmatrix} \tag{5.1}$$

and the differential movement from pose to pose provided by the dead reckoning and the absolute pose given by the MCL algorithm are used as measurements:

$$z_{odo} = \begin{bmatrix} dp_i \\ \vdots \\ dp_N \end{bmatrix} \; and \; z_{MCL} = \begin{bmatrix} \hat{p_1} \\ \vdots \\ \hat{p_N} \end{bmatrix}. \tag{5.2}$$

The errors to be minimized are:

$$\begin{cases} e^i_{MCL} = (x^i - z^i_{MCL}) \\ e^i_{odo} = (x^i - (x^{i-1} \oplus z^i_{odo})) \end{cases}, \tag{5.3}$$

Figure 5.2: Example of fitted trajectory vs. odometry trajectory

where superscript index means a component of respective vector and the operator $\oplus$ means transformation of x^{i-1} into x^i according to Equation 3.10. The minimized cost function is then:

$$J = \sum_{i=1}^{N} (e^i_{MCL} W_{MCL} (e^i_{MCL})^T + e^i_{odo} W_{odo} (e^i_{odo})^T), \tag{5.4}$$

where W_{MCL} is the weight of the MCL measurement and W_{odo} is the weight of the odometry. The weight for the odometry measurement was constant and the weight of the MCL measurement was obtained from the variance of the particle cloud.

An example of the outcome of the process described is illustrated in Figure 5.2. The original odometry is the white path, the MCL result is the green path, and the red path is the fitted reference trajectory. The computed trajectory is used as a reference path for the other measurements.

It must be noted that the reference trajectory is not the real trajectory. It is the best estimate of the trajectory that was achieved. Therefore the result should be considered as a deviation from the best estimate, because the reference is calculated from the same data as it is compared to.

Table 5.3: Different distances obtained with different methods

Set#	*time*	d_{real}	d_{lodo}	d_{SiLMU}
1	162.31	111.83	110.48	111.36
2	184.92	60.52	61.63	60.18
3	200.75	128.08	127.74	135.81
4	231.36	167.91	168.55	181.11
5	510.74	371.56	371.68	396.54
6	522.54	285.57	286.06	303.29
7	475.5	403.3	404.41	407.65
8	1208.3	584.92	592.6	600.91
Sum	3496.42	2113.69	2123.15	2196.85

5.2.2 Dead Reckoning Results

5.2.2.1 Case 1: Using pose space search laser scan matching

This subsection presents the results generated and published for a test report [117]. The dead reckoning output is calculated using SiLMU, the heading filter, and the pose space scan matcher algorithm. The dead reckoning output is obtained by using the translational output of the laser dead reckoning and heading from the heading filter. The trajectories were calculated in real time and were not post-processed afterwards.

The results are illustrated in Appendix A.1. Figures A.1, A.2, A.3 and A.4 plot the full route tracks with the laser scans integrated into the images. The laser points are plotted in the pose points obtained from the laser dead reckoning (red points). The SiLMU pose is illustrated in green.

Table 5.3 shows the different path distances obtained with different methods. In the table d_{real} is the distance obtained with the map-matching method, d_{lodo} is the distance obtained with the laser dead reckoning, and d_{SiLMU} is the distance obtained from SiLMU. The set number matches the numbers in Appendix A.1. Table 5.4 shows different errors. e_{lodo} and e_{SiLMU} are errors in meters when compared to d_{real}, $e_{start/end}$ is the distance error between the start point and the end point and e_{angle} is the heading error in degrees obtained as the difference in the heading between the last matched heading and the last heading returned by the laser odometry. Finally, Table 5.5 gives the relative errors.

As is visible from Table 5.5 the different approaches give different figures. The error in the distance travelled is well below 2% in all the laser odometry cases. The end-start error gives some idea about the error, but as can be seen from the figures in Appendix A.1, the end-start error does not reveal the true accuracy of the system.

Table 5.4: The dead reckoning errors

Set#	e_{lodo}	e_{SiLMU}	$e_{start/end}$	e_{angle}
1	1.35	0.47	2.7	-0.68
2	-1.11	0.34	1.07	1.82
3	0.34	-7.73	1.6	-1.3
4	-0.64	-13.2	3.78	-1.37
5	-0.12	-24.98	7.61	-8.49
6	-0.49	-17.72	13.87	-8.38
7	-1.11	-4.35	9.9	-0.27
8	-7.68	-15.99	30.42	-51.04

Table 5.5: Relative errors

Set#	$e_{lodo}(\%)$	$e_{SiLMU}(\%)$	$e_{start/end}(\%)$
1	1.21	0.42	2.50
2	-1.83	0.56	1.70
3	0.27	-6.04	1.30
4	-0.38	-7.86	2.20
5	-0.03	-6.72	2.10
6	-0.17	-6.21	4.90
7	-0.28	-1.08	2.45
8	-1.31	-2.73	5.10

5.2.2.2 Case 2: Using combined scan matching

The results presented in this subsection are obtained by calculating the differential movement between two scans from SiLMU and the heading filter and refining the pose estimate using Algorithm 5. These results are post-processed using the original PDR estimates generated for the test report [117]. The principal difference to the previous case is that there is no separate heading filter, which would provide a global heading. Instead, the heading is integrated from the output of the laser scan matching.

The results in Table 5.6 are presented in the same way as in the previous section. The full route tracks are illustrated in Appendix A.2. The figures are obtained by plotting the laser range scans into an occupancy grid for each calculated pose.

When the two cases are compared, the overall end-start error is similar in both cases. The heading error is also comparable, or even better in some cases when only laser dead reckoning is used. The comparison between the absolute path lengths differs. In Set No. 2 there is an unexplained 72-m path length measured for a distance that, according to the reference path, is approximately 60 m. Looking at Figure A.5 (in the middle), the difference is not too obvious. Now comes the question of what the real trajectory is. The method used earlier to obtain the reference path used laser dead reckoning from Case 1 as input, and therefore the reference trajectory is more likely to follow that trajectory. In other cases the distance error is considerably

Table 5.6: Results using combined scan matcher

Set	d_{lodo}	e_{lodo}	e_{angle}	$e_{start/end}$	$e_{start/end}(\%)$	$e_{lodo}(\%)$
1	116.79	-4.96	-2.65	2.6	2.32%	-4.44
2	72.48	-11.96	4.46	1.07	1.77%	-19.76
3	129.49	-1.41	8.82	0.88	0.69%	-1.1
4	169.75	-1.84	-8.54	9.27	5.52%	-1.1
5	373.08	-1.52	-0.21	2.69	0.72%	-0.41
6	302.59	-17.02	-31.57	8.21	2.87%	-5.96
7	418.47	-15.17	4.89	5.98	1.48%	-3.76
8	625.49	-40.57	-14.39	5.39	0.92%	-6.94

lower, but still higher than in the first case (which is probably explained by the same fact).

The combined scan-matching method provides some advantages over the previous one: 1) it is lighter in terms of the computational power required, and 2) it can provide an accurate heading estimate itself, which means that it can be used in combination with a MEMS gyro/IMU. As the results show (see Figures A.5, A.6 and A.7) the method performs well in all the test sets. Figure 5.3 shows the result of the coverage walk data set (Set No. 7). The outcome is an almost perfect map of the laboratory.

5.2.3 Map-matching methods

The map-matching methods were tested against the same data sets as the laser dead reckoning. The tests were performed with three different methods: 1) topological MCL only; 2) scan-based MCL, and 3) combined topo-scan-based MCL. All the methods were tested with a varying number of particles (from 100 to 20,000). The initial pose was read from the previously computed reference trajectory. The initial variance for the particle filter was given as $(1m^2, 1m^2, \frac{\pi}{30}rad^2)$.

There are two measures that can be used to compare the results: 1) the average of the variance of the particle cloud, and 2) the average deviation from the fitted path. The first measure tells how spread-out the particle cloud is. In general, the smaller the variance, the better the accuracy of the pose. The second measure shows the deviation from the reference path. For this one needs a true path to compare to. The method for generating the reference trajectories for laser dead reckoning provides a good result for comparing the path lengths, but it still might be inaccurate when it comes to pose-to-pose comparison. A map-based method was used to derive a reference path for the pose-to-pose comparison. The idea is to match the individual laser measurements to the map, which results in a reference trajectory that is not correlated with the data it is compared against (which was the case with laser odometry).

Algorithm 9 outlines the method used for matching the data to the map. The matching uses the MCL fitted paths to get the initial estimate of the real trajectory.

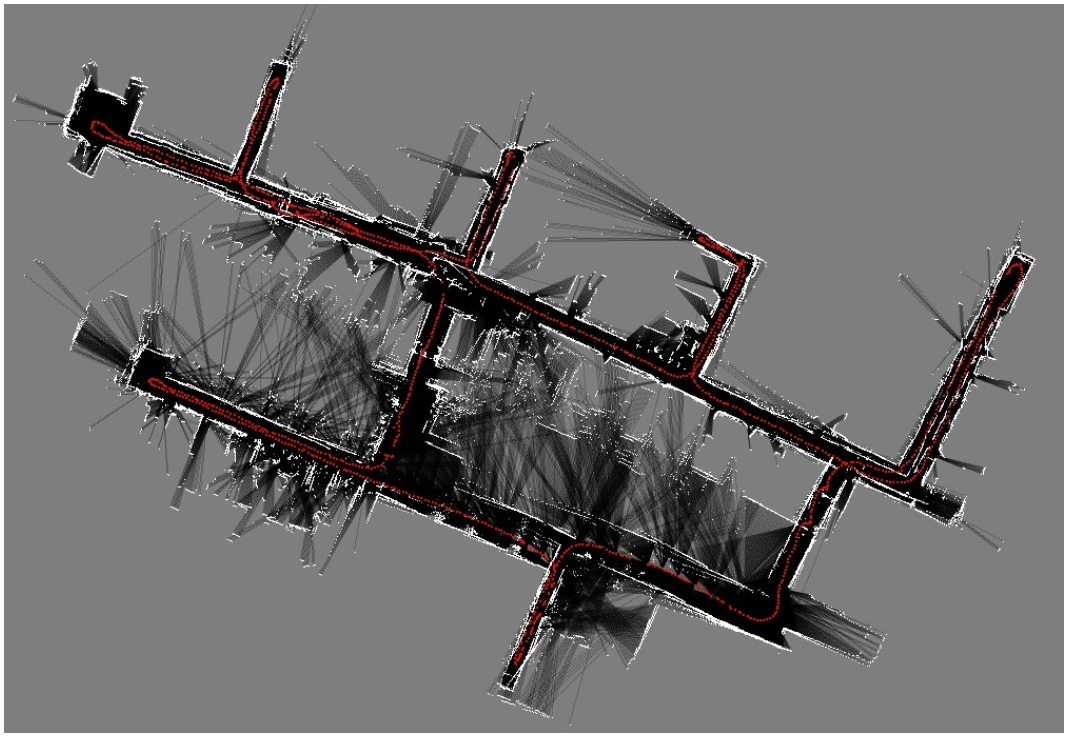


Figure 5.3: Result of coverage data set (set No. 7)

The paths are evaluated by calculating a correlation value for the whole trajectory (as the sum of the squared distance error between the scan points and map objects). After the selection of the best path from among the candidates, the correlation value is calculated for each individual pose. The poses are sorted according to the correlation value and matched to the map using a pose grid search (practically the same as the laser dead reckoning in Algorithm 4) to find the pose with the maximum correlation. After matching the laser measurement is added to the map. The main reasons for this process are: 1) the poses that correlate best most probably produce correct matches, and 2) the process reduces the correlation between the poses; that is, if the poses were matched sequentially, the previous estimates would affect the estimation of future poses.

An example of the path (and map) is given in Figure 5.4. The path is close to the correct one on average, but there are also a few false matches. Furthermore, the path that the method outputs is not smooth. Each pose is matched individually to the map, without the dead reckoning estimate being considered. Therefore this method was not used when the lengths of the dead reckoning trajectories were compared.

Tables B.1-B.8 in Appendix B show the results from different runs. Each data set is in a separate table. The table is organised so that first there are the results of topo-MCL, then scan-MCL, and finally topo-scan-MCL. In the table Var(x), Var(y), and Var(a) show the mean variance of x, y, and heading, and err(x), err(y), and err(a) show the average deviation from the reference trajectory. The same runs are depicted in Figures B.1-B.8. In the figures the trajectories of all the runs are plotted into the

Algorithm 9 Correlation-Based Map Matching

Inputs: Map, Laser measurements and N computed MCL trajectories

1. Compute correlation value for all N paths
2. Select the best path P
3. Compute the correlation for each pose in P
4. Sort P according to obtained correlation value
5. Go through all the poses starting from the one that correlates best
   (a) Fit the measurement to the map
   (b) Add the measurement to fitted pose in the map

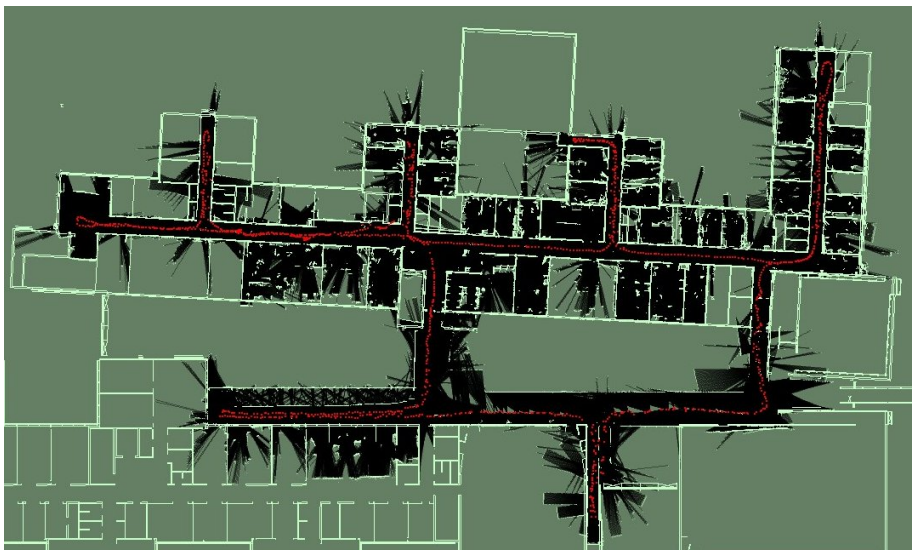


Figure 5.4: Map-matched trajectory for data set No. 7

Table 5.7: Run times of different algorithms [ms/measurement]

Particles	Topo	Scan	Topo-Scan
100	0.659	1.542	0.891
200	0.753	2.298	1.071
500	1.077	4.831	1.660
1000	1.556	8.621	2.600
5000	5.910	39.379	10.203
10000	11.281	79.203	19.700
20000	21.989	155.589	38.514

same picture.

Table 5.7 shows the different run times of the algorithms. The unit in the table is milliseconds per measurement. Not surprisingly, the topo-MCL algorithm is the fastest. The difference between scan-MCL and topo-scan-MCL is explained by the fact that the measurement likelihood of scan-MCL is updated with every measurement, while with topo-scan-MCL the measurement is only updated if there has been enough movement between the last update and the current pose (to prevent the bias from an incorrect map). Table 5.7 shows that basically all the algorithms can be run in real time with 5000 particles.

A few failures occurred during the test runs. Scan-MCL caused failure in Test Set No. 5 with 5000 particles. During Set No. 8 all methods failed with 100 particles. This is not surprising, since 100 particles are not able to represent distributions well. However, there does not seem to be any particular reason for the failure of scan-MCL during Set No. 5, except bad luck. The scan-MCL algorithm updates the measurement likelihood each time the measurement arrives. This, on some occasions, may cause bias in the particle distribution, since the map is not actually representing the current state of the environment. If the biased distribution is updated with the SIR algorithm, it is possible that none of the remaining particles will describe the true distribution of the pose. Table 5.8 shows the results of a repetition trial. Each method was run a hundred times using 100, 200, 500, and 1000 particles. The ratio in the table indicates the number of failures. Clearly, topo-MCL provides better results when more particles are added. Scan-MCL provides the best results overall, but there are four failures when 1000 particles were used, while there were zero failures with 500 particles. Topo-scan-MCL also requires more particles by nature, but it seems to provide consistently better results when particles are added. The topo-MCL and topo-scan-MCL methods were also tested with 5000 particles. In both cases the result was zero failures.

In most cases topo-MCL has the largest variance and the worst pose estimate. This is not a surprising result. In fact, the accuracy of topo-MCL depends on the environment and the type of movement. For example, when a human is walking in a corridor, the particle cloud will eventually be at least the size of the width of the corridor. On the other hand, when the trajectory goes through narrow spaces, such as doorways, the distribution will get smaller. In most cases scan-MCL has the

Table 5.8: Results of repetition trial

Method	100	200	500	1000
topo-MCL	82/100	43/100	14/100	1/100
scan-MCL	7/100	2/100	0/100	4/100
topo-scan-MCL	36/100	14/100	2/100	0/100

smallest variance, but topo-scan-MCL is the most accurate. This is explained by the fact that scan-MCL updates the measurement likelihood whenever a new scan has arrived. Updating the measurement likelihood causes more variance in the particle probabilities, which causes the triggering of the SIR update more frequently than in the case of topo-scan-MCL. Table 5.9 shows the comparison of the methods when 1000 particles were used. Each group of three has its own data set and the order is the same as before. The italics are used to represent the best method for each of the cases.

5.2.3.1 Localisation without environmental perception

The tests in the previous sections used a laser-based dead reckoning estimate for map matching. Conceptually there is no difference between the estimates, but to prove that topological MCL could be used without any environmental perception sensor, the tests were performed with a PDR estimate only (i.e. SiLMU and a heading filter are used for position estimation and topo-MCL for map matching). The only difference in implementation to the previous runs was that the variance of measurement was increased.

Figure 5.5 shows the results of the three most challenging data sets: 1) the room search (Set No. 6); 2) the coverage walk (Set No. 7), and 3) the automation laboratory walk (Set No. 8). In all cases the method is able to track the pose successfully. Similarly to other cases, a repetition trial using data set No. 8 (see Table 5.10) was run to see the effect of the number of particles and the reliability of the method. Using an "unreliable" dead reckoning estimate makes it necessary to use large variance for the noise (even as much as a standard deviation of 40% of the distance travelled). Naturally, this has effects in two ways: 1) the weighted average is not as accurate an estimate of the true pose as in previous cases, and 2) the filter requires more particles to represent the pose distribution. Table 5.10 shows that in this case 5000 particles were required to achieve a robust estimate, while earlier about 1000 were enough. Nevertheless, the computational load of the method allows even more to be used. When 5000 particles are used, the method runs for 17.8 seconds (using Intel(R) Core(TM)2 CPU 6300 @ 1.86 GHz) for the whole of data set No. 8, which in real time took 1208.3 seconds.

Table 5.9: Comparison of methods with 1000 particles

Method	Var(x)	Var(y)	Var(a)	err(x)	err(y)	err(a)
topo-MCL	2.6	0.83	0.01	-0.28	-0.02	-0.01
scan-MCL	*0.58*	*0.08*	*0*	0.25	0.11	-0.02
topo-scan-MCL	0.66	0.12	0	*-0.09*	*-0.06*	*-0.01*
topo-MCL	0.89	0.66	0.03	0.42	-0.05	-0.02
scan-MCL	*0.18*	*0.09*	*0.01*	*-0.04*	*0.03*	*-0.02*
topo-scan-MCL	0.28	0.13	0.01	0.08	0	-0.02
topo-MCL	0.59	0.69	0.11	*0.01*	*0.05*	*-0.01*
scan-MCL	*0.29*	*0.23*	*0.03*	0.05	0.14	-0.01
topo-scan-MCL	0.3	0.22	0.04	0.09	0.01	-0.01
topo-MCL	1.11	0.71	0.01	-0.58	-0.03	0.03
scan-MCL	*0.17*	*0.09*	*0.01*	-0.05	0.03	-0.03
topo-scan-MCL	0.33	0.16	0.05	*-0.05*	*-0.02*	*-0.01*
topo-MCL	0.46	0.66	0.04	-0.01	0.16	0.03
scan-MCL	*0.16*	*0.14*	*0.02*	0	0.05	0
topo-scan-MCL	0.26	0.17	0.08	*0.01*	*0.03*	*0*
topo-MCL	0.39	0.53	0.07	-0.13	-0.23	-0.01
scan-MCL	*0.17*	*0.09*	*0.03*	-0.07	-0.02	0.01
topo-scan-MCL	0.16	0.12	0.06	*-0.02*	*-0.02*	*0*
topo-MCL	0.8	1.01	0.03	-0.04	0.07	0.02
scan-MCL	*0.31*	*0.2*	*0*	-0.12	-0.13	-0.06
topo-scan-MCL	0.56	0.4	0.01	*0.01*	*0*	*-0.01*
topo-MCL	0.34	0.4	0.04	-0.03	0.02	0
scan-MCL	*0.09*	*0.09*	*0.01*	-0.06	0.05	-0.01
topo-scan-MCL	0.13	0.1	0.01	*-0.02*	*0.01*	*-0.02*

Table 5.10: Repetition trial for dead reckoning-only localisation

Particles	1000	2000	5000
Error rate	29/100	4/100	0/100

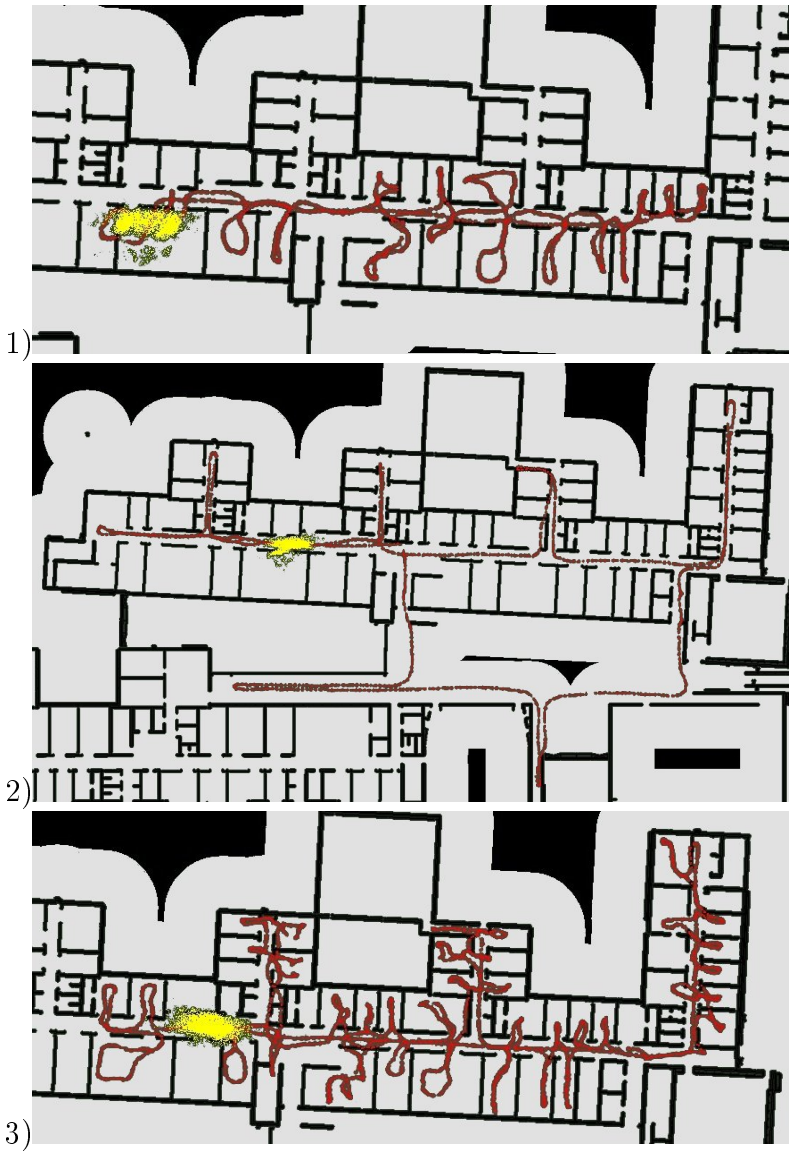


Figure 5.5: Results of localisation without laser

Table 5.11: Results of map sensitivity repetition trial (failures out of 50 repeats)

	topo-MCL	scan-MCL	topo-scan-MCL
Remove 1x(6mx6m)	0	8	0
Remove 2x(6mx6m)	0	5	0
Remove 3x(6mx6m)	0	4	0
Remove 4x(6mx6m)	0	3	0
Remove 5x(6mx6m)	0	7	0
Remove 6x(6mx6m)	0	1	0
Remove 6x(8mx8m)	2	45	0
Remove 6x(10mx10m)	11	50	18
Remove 6x(15mx15m)	50	50	48
Add 5 walls	1	28	0
Add 10 walls	5	44	2
Add 15 walls	11	47	7
Add-5-Remove-3	2	33	3
Add-10-Remove-5	16	44	9
Add-10-Remove-10	33	50	35

5.2.3.2 Sensitivity to map errors

The Monte Carlo Localisation method requires a map as a reference. The map used in the previous sections is based on the CAD model of the building. All the major structures (outer and inner walls) are on the map. In a sense the map is a rough representation of the environment, because it lacks all the furniture etc. On the other hand, the CAD map is (at least from the topological point of view) a very accurate representation of the environment. The question is whether these methods could be used with a less accurate map. This was tested by: 1) removing parts of the map; 2) adding extra walls to the map, and 3) removing parts and adding walls and running repetition trials for the methods. These tests were tested against the automation laboratory walk (Set No. 8).

The results are summarized in Table 5.11. Each method was run fifty times with one set of parameters. After each run the map was regenerated. The "Remove" tests had a random start position for the block that was removed (somewhere on a line which was on the trajectory). The "Add wall" tests randomly generated the walls into the map.

The results show that topo-MCL and topo-scan-MCL are relatively insensitive to missing walls. There are no errors until six eight-by-eight-metre blocks have been removed from the map (after this, most of the upper part of the map is missing).

Adding extra walls causes more errors in the trials. This is mostly due to the fact that when a trajectory passes an extra wall, the distribution is practically initialised (all particles will result in the same probability). In all cases scan-MCL is the most sensitive to map changes. Scan-MCL updates the distribution only on the basis of the sensor readings and therefore an inconsistent map causes the most inconsistency

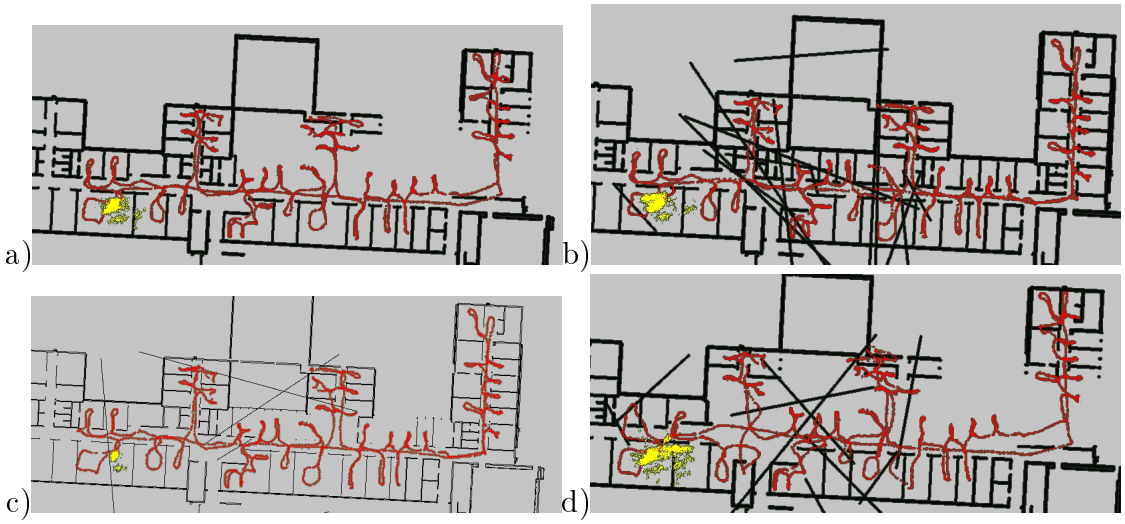


Figure 5.6: Example images from map correctness tests: a) topo-MCL with six ten-by-ten-metre blocks removed; b) topo-MCL with 15 added walls; c) topo-scan-MCL with five added walls and five six-by-six-metre blocks removed; d) topo-MCL with ten added walls and six ten-by-ten-metre blocks removed.

in the pose distribution.

Figure 5.6 shows some example images from the trials. The images show that the methods are able to recover from significant differences in the map. Nevertheless, adding extra walls affects the estimation and may result in a wrong pose estimate.

5.2.3.3 Reference test with VTI Indoor Pedestrian Navigation demonstrator

VTI Technologies has developed a Pedestrian Navigation demonstrator to demonstrate their new 3-axis accelerometer (CMA3000). The demonstrator was first introduced at the Consumer Electronics Show (CES) in Las Vegas on 8.-11.1.2009. The author had an opportunity to test the device and these tests are reported here. The purpose of the tests is to show that the method (namely topo-MCL) can be applied to other devices than those reported in this thesis. VTI's demonstrator also represents a different type of PDR device than those presented in this thesis and it uses components that will potentially be found from any mobile phone in the future.

The device consists of a chest-mounted module (see Figure 5.7) and a host computer. The chest-mounted module uses a 3-axis accelerometer, one axis gyro, and a three-axis compass (which was not used in these tests) to derive the speed, distance, and heading of a pedestrian. The speed, distance, and heading information is sent to the host computer through Bluetooth. The host computer calculates the pose and displays it on a user interface. In these tests the pose was recorded into a log-file and later processed with the topo-MCL algorithm.

The speed and distance calculation is based on the same algorithms as used with sport pods. The algorithm was developed for walking and for running. Basically,

Figure 5.7: VTI's Pedestrian Navigation Module

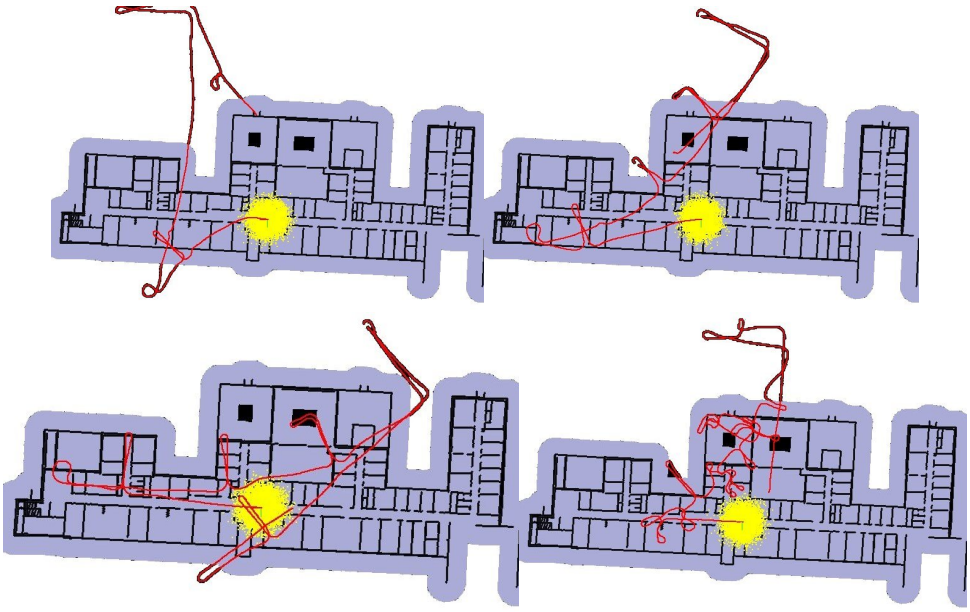


Figure 5.8: Dead reckoning paths obtained from VTI's demonstrator plotted on a map.

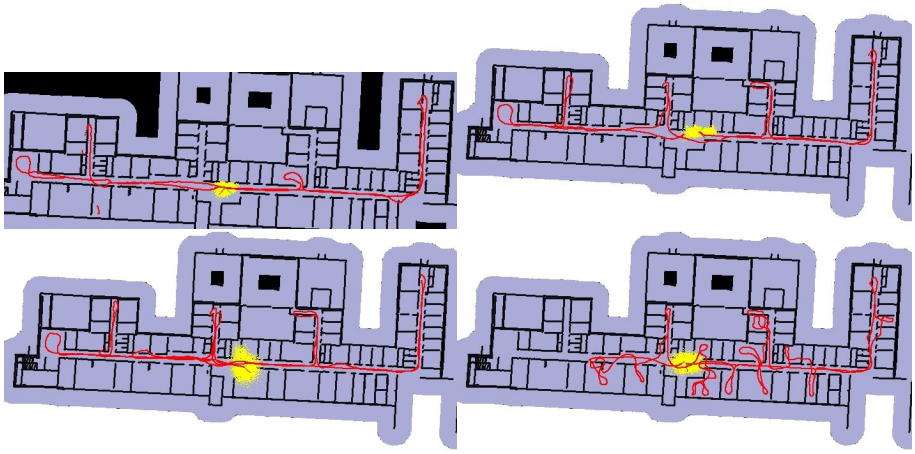


Figure 5.9: The results after map-matching

the algorithm is able to compute the speed of a pedestrian with varying step lengths, but it requires constant walking (or running). For example, if the pedestrian stops and then starts walking again, the device loses a couple of steps (at least in the current implementation).

In this case the device was tested "nicely" so that the tests were performed by walking, and the walking pattern was kept relatively constant (except that in data set No. 2 there was a stop caused by opening a door and in data set No. 4, where the trajectory goes through several rooms). The dead reckoning paths are illustrated in Figure 5.8. In all the cases there is a significant heading error (e.g. the first data set has a heading error of almost 90 degrees within the first 30 m of walking). The distance estimate is relatively consistent, but because the device does not estimate the trajectory of the sensor (it estimates the speed on the basis of accelerations), it requires some calibration. In this case the distance between two consecutive poses was multiplied by 1.1 to obtain a more accurate path length.

The results after the application of topo-MCL to given data sets are shown in Figure 5.9. In all the cases the trajectories are corrected to follow the true trajectory. For the first data set (upper left-hand corner in Figures 5.8 and 5.9) the algorithm had to be tuned quite a number of times before it was successful (noise parameters and selection of the map). From the initial pose the gyro drifts rapidly and there are two possible trajectories (one outside the laboratory and one along the corridor) which caused failures for the estimation. Nevertheless, the tests show that the VTI's demonstrator (and especially the principle it uses) could be used for indoor localisation with the topo-MCL algorithm.

5.2.3.4 Towards 3D indoor localisation

All the work and results presented so far have been restricted to 2D. In reality buildings are rarely in one plane and as a natural extension of the work this section presents preliminary proof-of-concept results on 3D personal localisation. The use

Figure 5.10: Microstrain's Inertial-Link mounted on a foot

of a foot-mounted IMU has been shown to be a solid way of obtaining the 3D trajectory of a pedestrian. Following the work done in [54, 104, 105, 8], simple PDR system was put together, which outputs the 3D pose. The setup consists of an IMU (Microstrain Inertia-Link), which measures 3D acceleration and 3D angular rates. The data are collected by mounting the IMU into a shoelace (see Figure 5.10) and connecting the device to a laptop, which collects the data.

Inertia-Link has its own internal orientation estimation and it can output an orientation matrix with the acceleration and angular rate data. The sensor's orientation estimation was found to be accurate enough if the walking speed was not too high and it was used to transform the accelerations into an Earth frame of reference. The 3D trajectory of a walker is obtained by double-integrating the transformed (and gravity-corrected) accelerations. A zero-velocity update is applied to the estimated speeds every time a foot is considered to be on the ground. This was determined by using a fixed threshold for the acceleration magnitude combined with angular rate magnitude.

The dead reckoning calculation is performed with an Octave script, which records the output in a file. An example of a dead reckoning path is illustrated in Figure 5.11. The result is comparable to the results obtained with laser dead reckoning. However, it must be underlined that the tests are preliminary and required a relatively slow walking speed. Walking that is too fast confuses the internal orientation estimation of the sensor, which causes the pose estimation to fail.

The map matching as presented in the thesis is based on 2D maps. Extending the methods to full 3D would require 3D maps of buildings, which in general are not available. Instead, the estimation in this test is divided into floors and a separate estimation of height. The estimation is initialised to a certain floor. The 2D pose estimation is made on that floor and the z-coordinate is assumed to be static while on the floor. The stairs are detected by comparing the difference in the z-coordinates between consecutive steps. When the z-coordinate is close enough to a new floor the particles from the previous floor are copied to the new floor and the estimation

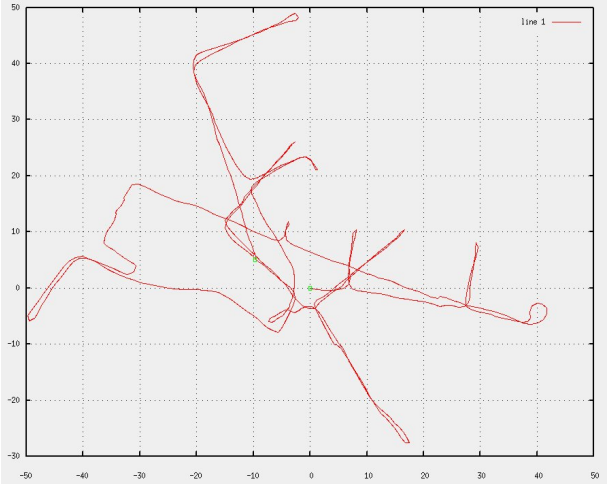


Figure 5.11: Dead reckoning path obtained with Inertia-Link. Length of the path is approximately 630m

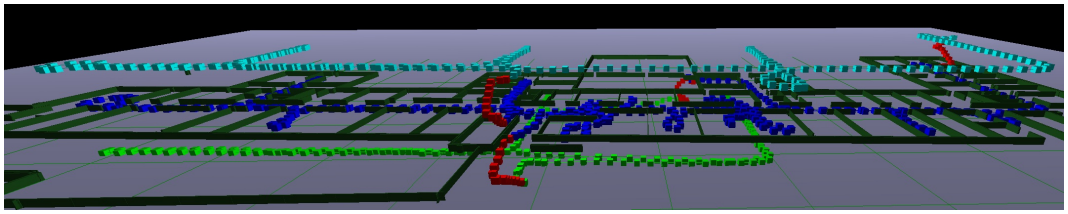


Figure 5.12: Result of map-matched trajectory that is approximately 965 m long and goes through three floors. The different colours represent different floors and red represents stairs. The second-floor map is drawn as a reference to the image.

then continues on the new floor.

Figure 5.12 illustrates a test walk that goes through all the corridors on all the three floors of the TUAS building. The different colours of the boxes represent steps taken on different floors. The red colour is used for steps taken on stairs. Only the second-floor map is plotted as a reference (plotting the maps of all the floors confuses the image).

5.3 Human-Robot Team Tests

PeLoTe demonstrated a teleoperated human-robot team performing a simulated fire-fighting mission. The PeLoTe system is designed to support the coordination of multiple heterogeneous entities. The design of the system is introduced in greater detail in the recent PhD thesis by Frauke Driewer [36]. The complete design process includes [36]:

- finding SA requirements for the team and for the team members;

- the design of supervisory functions, which includes how the task is divided and monitored and when intervention is required;
- the design of the autonomy level of the system (robot autonomy and team autonomy support);
- designing interfaces that allow humans to perceive, understand, and modify the necessary information.

Finding the SA requirements is essentially finding what information is relevant during the mission and for whom. This can be done by analysing the task and by end user analysis [33]. In this case the task is essentially an exploration of a (partly) unknown environment. Additional constraints come from the end application (fire-fighting), which additionally adds some critical requirements concerning the safety of the team and "victims" (e.g. knowledge of the location of the fires and other dangerous places). The team SA in the case of PeLoTe is maintained through a common model. The common model integrates all the SA elements from all entities and represents them in a single frame of reference. The relevant objects, space, and task state are modelled as abstract items which can be shared with different entities through their own interfaces. The interface means in this case a transformation into an entity-understandable form (an object is displayed to a human through GUI, while for a robot it may only be a labelled location). A more detailed description of the SA requirements of the team and team members can be found from [36].

The supervisory functions in this case are to control the team by providing inspection points (or exit points), to monitor the performance of the task, and to modify the ongoing plan if required (e.g. if a robot found a victim, the operator would command a human entity to escort him/her to the nearest safe exit). Additionally, the operator was mapping the observations coming from the robots (or sometimes from humans) onto the map. Basically, the robots reported an event if there was a potential observation to be added to the map and the operator confirmed this by observing the sensor data coming directly from the robots.

The autonomy level of the PeLoTe system can vary from direct teleoperation to full autonomy. In normal operation the robots, as well as the human entities, follow given trajectories and make observations autonomously. The robots could be put under direct teleoperation if required. Obviously the human entities represent full autonomy, but even they were expected to follow given instructions. To support the team's autonomy there was a planning system which could be used to give trajectories to all entities [82].

The graphical user interface is based on the common model. The information from the model is displayed in different layers, which allows the operator to filter out irrelevant information if necessary. The GUI displays the information with respect to the geometric map. The GUI is operated in a point-and-click manner. For example, giving a plan to an entity requires only the selection of the entity, choosing "plan to", and clicking the target. An example of the operator's GUI and the human entity's GUI is given in Figure 5.13. GUIs are based on the same framework, but

the differences are that the operator's GUI has an overview window and entity control panels, while a rescuer's GUI has a display which shows the sensor data around him/her. The GUI was designed with several iterations. First, an end user requirement analysis was performed on the basis of interviews conducted by sending questionnaires to several rescue organisations. The first prototype was based on the analysis. Later the GUI was demonstrated and evaluated by fire-fighting professionals. More information about the GUI and its development can be found from [35, 32, 33, 34, 36].

The overall schematics of the PeLoTe system are illustrated in Figure 5.14. Basically, all information is shared through the common model. Additionally, the operator has a voice channel to the human entity and, if required, may teleoperate the robot entity. In the nominal case the operator perceives the mission through the common model. The remote entities travel in a real workspace and report their state and findings with respect to the common model. Thus, in order to maintain a consistent common model, the information produced from the real environment must match the common model. In other words, the information must be provided in a correct frame of reference. This is why the localisation system for a human is of great importance in this type of system.

The whole PeLoTe system has been put together on three occasions. The first experiment (called the semi-final experiment) was performed on October 21-22, 2004 in the Physics Building of the University of Würzburg. The first system prototype was successfully integrated and tested. Nevertheless, both the system and the experiment design still showed some drawbacks. On the basis of the results of these experiments, the system was improved. On November 20-21, 2004 the test was repeated in the final experiment. The Computer Science building of the University of Würzburg was used as a test area. Half of the test participants were volunteer fire-fighters. On November 18-19, 2004 the system was tested and demonstrated under more realistic conditions in the Würzburg fire training house. The fire-fighters of the training house observed the tests. The purpose of the demonstration was to get feedback from the potential end users of such a system (see Figure 5.15).

The purpose of the experiment was to measure the effect of using the telematic system in the coordination of a remote team. The PeLoTe system was evaluated with several questionnaires to evaluate the operator's situational awareness, as well as its negative and positive effects. The performance was measured by calculating several performance values during the mission, such as completion time, area coverage, victims found etc. From this point of view, PeNa was just one part of the system. The evaluation of the influence of PeNa on the whole system from the system evaluation is difficult. Nevertheless, the system tests are presented as proof-of-concept results of using PeNa for incorporating the human entity into the system.

A complete evaluation of the PeLoTe experiment has been published in [35, 125, 109]. Moreover, in her PhD thesis Driewer [36] analyzes the experiment results critically from the user's point of view on the basis of interviews with the test persons.

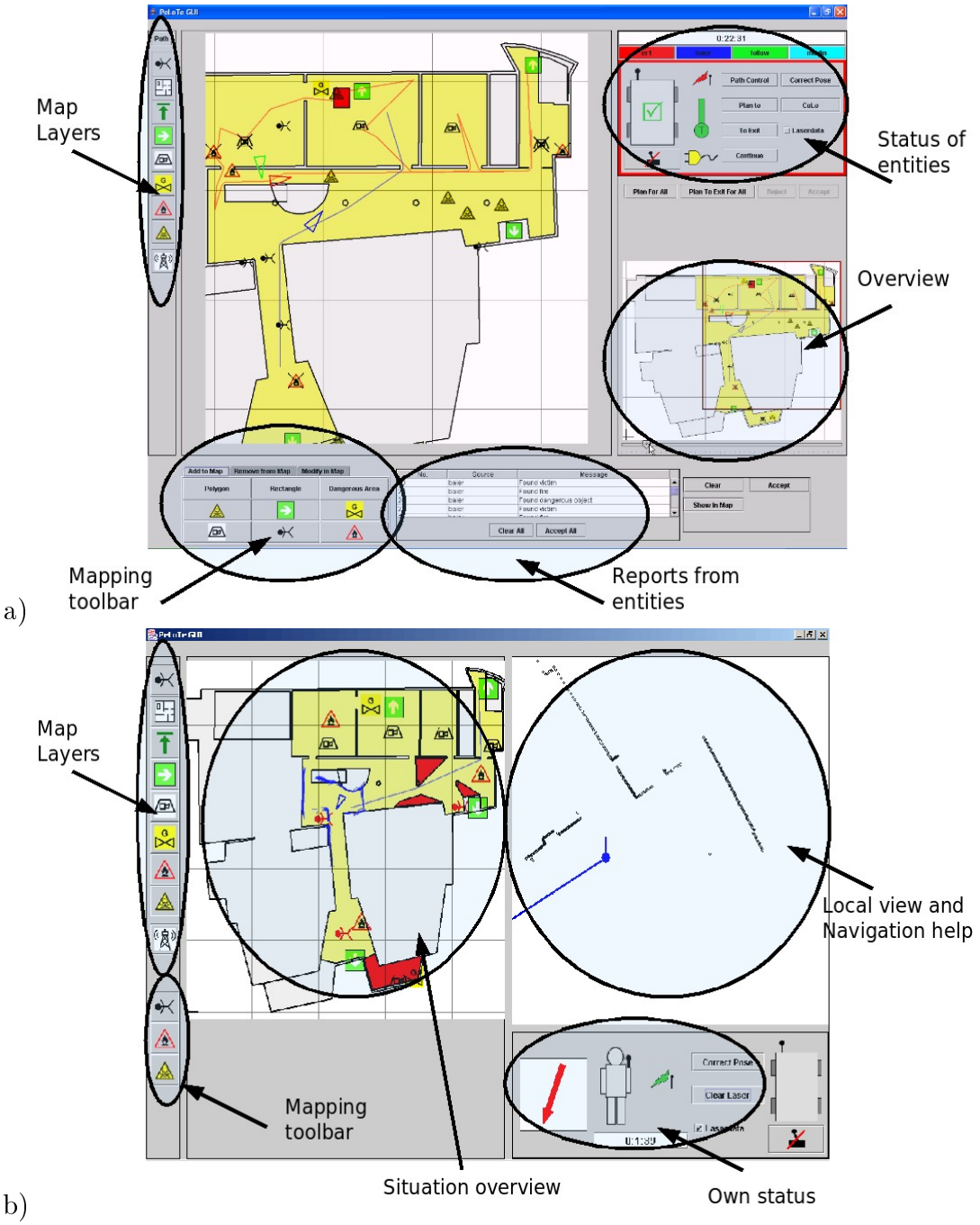


Figure 5.13: Examples of the user interfaces. a) the GUI for an operator b) the GUI for the rescuer

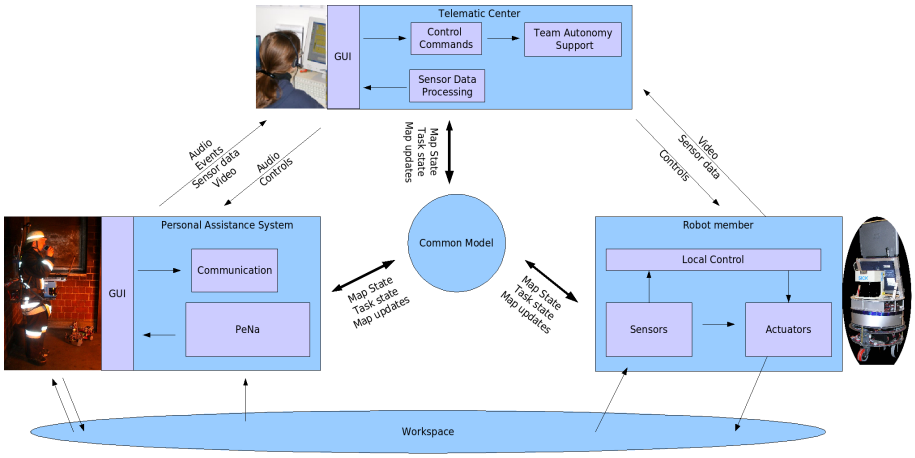


Figure 5.14: PeLoTe system schematics

Figure 5.15: End user demonstration

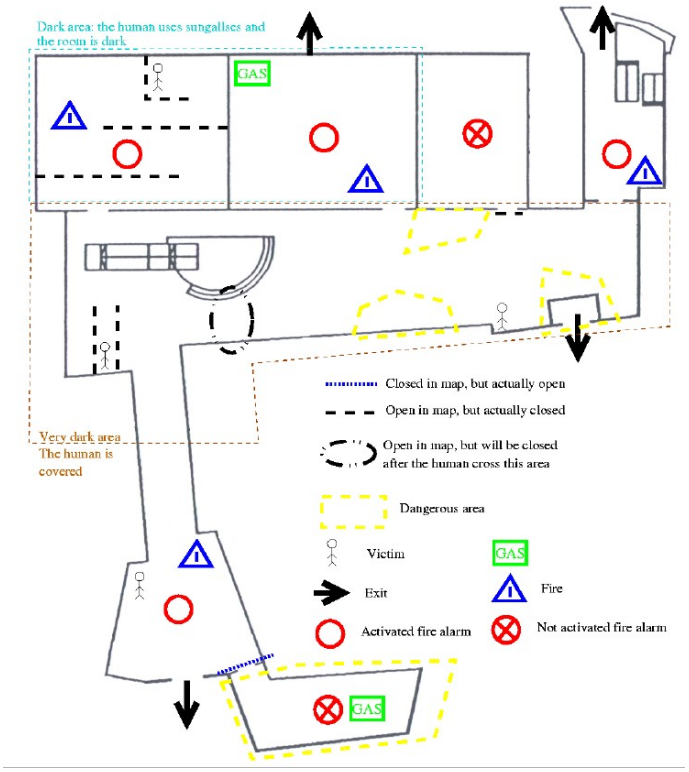


Figure 5.16: Experimental design

5.3.1 The experimental setup

The experiment was performed in the basement of the computer science building. It provided an open space, as well as rooms to inspect. In order to make the area more challenging and unknown for people who were familiar with this building, changes, such as additional obstacles and a maze, were built into the environment. Figure 5.16 shows the experimental setup.

Victims, simulated by crying dolls, needed to be rescued and taken to a safe exit. Symbols in the area represented fires, dangerous objects, gas valves, fire detectors, and emergency exits. Fires had to be put out by touching the symbol. Fire detectors had to be checked to see if they were activated or not. Structural collapses were simulated by obstacles being moved during the test run. Certain areas were blocked by cordons symbolising dangerous areas. Human rescuers were not allowed to enter these areas. Some areas were made dark by darkening the windows. These areas simulated low-visibility areas. In some parts of the environment, a blanket was used to cover the participants, which simulated no-visibility conditions.

Alternately, a team with the PeLoTe system and a team without the system were sent to fulfil the mission. A team consists of two people, one supervisor and one human rescuer in the emergency area. Additionally, the PeLoTe team had one robot that was exploring the area on its own and one robot that was following the human entity. The follower robot was used for direct (joystick) teleoperation in the areas the human entity could not go into. The teams were trained to use the system in

Table 5.12: Performance of the traditional teams

Team	Time [min]	Coverage	Fire	Danger	Victim
1	26	80	3	3	4
3	23	95	3	5	2
5	21	95	4	4	4
7	18	80	2	3	4
9	25	95	3	4	4
11	28	95	4	5	3
Average	**23,5**	**90,0**	**3,2**	**4,0**	**3,5**
Day 1	23,3	90,0	3,3	4,0	3,3
Day 2	23,7	90,0	3,0	4,0	3,7

different corridors for about 30 minutes. It was found during the tests that the 30-minute training was insufficient for most of the teams. This was visible in the form of insecure behaviour with the system, as well as from the discussion between the operator and the human entity. This alone causes divergence in the results and weakens their reliability.

The teams without the PeLoTe system had a paper map and audio communication available. They were also carrying a backpack with an additional load for the purpose of comparison with the current weight restrictions of the localisation system. The given map was only partly correct. The test teams were instructed to observe a time limit of approximately 25 minutes (no strict time limit was given).

5.3.2 Results

The experiments verified that all the components were working as expected, and that the system could be used by other people than the developers. The test participants were students, staff, and volunteer fire-fighters, i.e. they had very different experience with computers (from 1 hour a week to 50 hours a week). Nevertheless, they were all able to use the system after a relatively short training period.

Table 5.12 shows the performance of the traditional teams, as well as an average over all the teams and the teams on the first and the second day, i.e. non-fire-fighters and fire-fighters. Table 5.13 shows the same contents for the PeLoTe teams. The last column shows how often the rescuer in place used the following robot.

On average the PeLoTe teams performed better than the traditional teams. They rescued all the victims, put out more fires, found more dangerous areas, and explored the area more completely. On average the traditional teams performed faster, but this was mainly because the PeLoTe teams were teleoperating the robot during the mission, which took a lot of time. Therefore the time criterion had a low priority.

Table 5.14 shows the ranking of all the teams on the basis of the performance evaluation. Team 2 performed excellently: they rescued all the victims, put out all the fires, and found all the dangerous areas. Moreover, they covered the area completely and performed the task in below 25 minutes. Team 12 performed equally well, but they needed a little more time. Both teams also used the robot to explore the dangerous area, which otherwise would not have been accessible. The next six ranks are occupied by four PeLoTe teams and two traditional teams, which all

Table 5.13: Performance of the PeLoTe teams

Team	Time [min]	Coverage	Fire	Danger	Victim	Robot
2°	24	100	4	5	4	1
4°	22	100	3	5	4	2
6^	25	95	4	3	4	1
8	23	100	3	5	4	0
10	29	95	3	4	4	1
12	28	100	4	5	4	2
Average	**25,2**	**98,3**	**3,5**	**4,5**	**4,0**	
Day 1	23,7	98,3	3,7	4,3	4,0	
Day 2	26,7	98,3	3,3	4,7	4,0	

° Teams that had used the PeLoTe system already in the semi-final experiment
^ GUI of human in place did not receive updates

Table 5.14: Ranking of teams on the basis of performance evaluation

Rank	Team	Rank	Team	Rank	Team
1	2°	5	4°	9	1
2	12*	6	8*	10	7*
3	5	7	9*	11	11*
4	6^	8	10*	12	3

* Teams day 2 (fire fighters)
^ GUI of human in place did not receive updates
° Teams that had used the PeLoTe system already in the semi-final experiment

performed equally well. All six teams rescued all the victims, put out 3 to 4 fires, found 3 to 5 fires, and covered 95% to 100% of the area. Team 6 had a system failure, which prevented the human GUI from receiving updates from the operator. Nevertheless, the supervisor was able to track the position of the human from the GUI and could guide the human entity by audio communication.

Table 5.15 the details of the memory test. The memory test was used to evaluate how well the operator and the human entity remembered the mission afterwards. The task was to place all the items and exceptions on the map. The numbers in Table 5.15 are the percentages of recalled objects from the found or mapped objects (recalled object * 100% /objects).

The different objects or events in Table 5.15 are: location of victims; location of found fire; location of dangerous areas, and state of alarms. The symbols Map1 and Map2 in Table 5.15 indicate the changes to the map. Map1 are places that were marked as accessible on the map, but actually were not. Map2 was the place that closed behind the human during the mission (simulated structural change).

Table 5.15 points out the differences in the memory test between the supervisors and human entities for both teams without and with the system. Without the system the human entity and the supervisor could remember the situation equally well. When comparing individual items the human entity was able to remember the events slightly better. This indicates that the supervisor had difficulties in understanding the situation in the emergency area.

Table 5.15: Results of memory test, comparing supervisor and human entity for teams without and with system

		Victims	Fire	Danger	Alarms	Map1	Map2	Average
without system	supervisor	61	60	33	42	4	33	**39**
	in place	69	71	37	42	13	0	**39**
with system	supervisor	79	63	77	75	24	50	**61**
	in place	83	76	42	39	34	17	**49**

Table 5.16: Memory test results from the semi-final experiment

		Victims	Fire	Danger	Alarms	Gas	Map	Average
without system	supervisor	75	50	83	90	71	52	**70**
	in place	79	64	83	62	43	73	**67**
with system	supervisor	71	57	100	33	43	32	**56**
	in place	79	67	100	43	43	61	**65**

With the PeLoTe system, both the supervisor and the human entity could remember the situation better than the participants in the control group. Table 5.15 shows significant differences between the supervisor and the human entity. On average the supervisor was able to remember the events during the mission better. When individual items were being evaluated it could be seen that the human entities remembered victims, fires, and places that were marked as accessible on the map, but were actually closed, slightly better. This means that the human entities could remember an object better if they were more in contact with it. Basically, the difference is that the operator has a better overview of the whole situation and that the operator was receiving information on events from both the robot and human entity (difference between individual SA and team SA).

It was also observed during the experiment that the groups without the system felt more negative feelings than the groups with the system. Supervisors without the system became nervous fast since they lost track of their fire-fighter in the house. They tried to understand what was happening inside the area, but since they could only guess or sometimes had no clue at all, they very quickly started to feel helpless and not needed. The human entitities without the system felt alone and often repeated that they were lost. The communication between both team members was often louder and more stressed than with the PeLoTe teams. The supervisors with the system were observed to be more relaxed.

Overall all the indicators showed that the PeLoTe system improved the understanding of the situation for the whole team. This is not surprising as the shared model updates the events in real time for all. Thus, as long as the added information is correct, the model represents the situational view well.

One question is how important the correct positioning is in the creation and maintaining of situational awareness. One indicator with regard to this question is obtained by comparing the results of the semi-final experiment to the final experiment. A total of 14 teams participated in the semi-final experiment. All the teams filled the same questionnaires as in the final experiment. The setup was similar to the final experiment, but took place in a simpler environment (only corridors were included). Table 5.16 shows the results of the memory tests. The result is quite surprising, since it differs completely from the one obtained from the final experiment. The memory test showed that the control group had a better understanding of the situation.

When the reason for this was being analysed, one obvious observation was that the system did not function properly. During the semi-finals the PeNa localisation was not based on MCL, which resulted in the PeNa system needing manual position

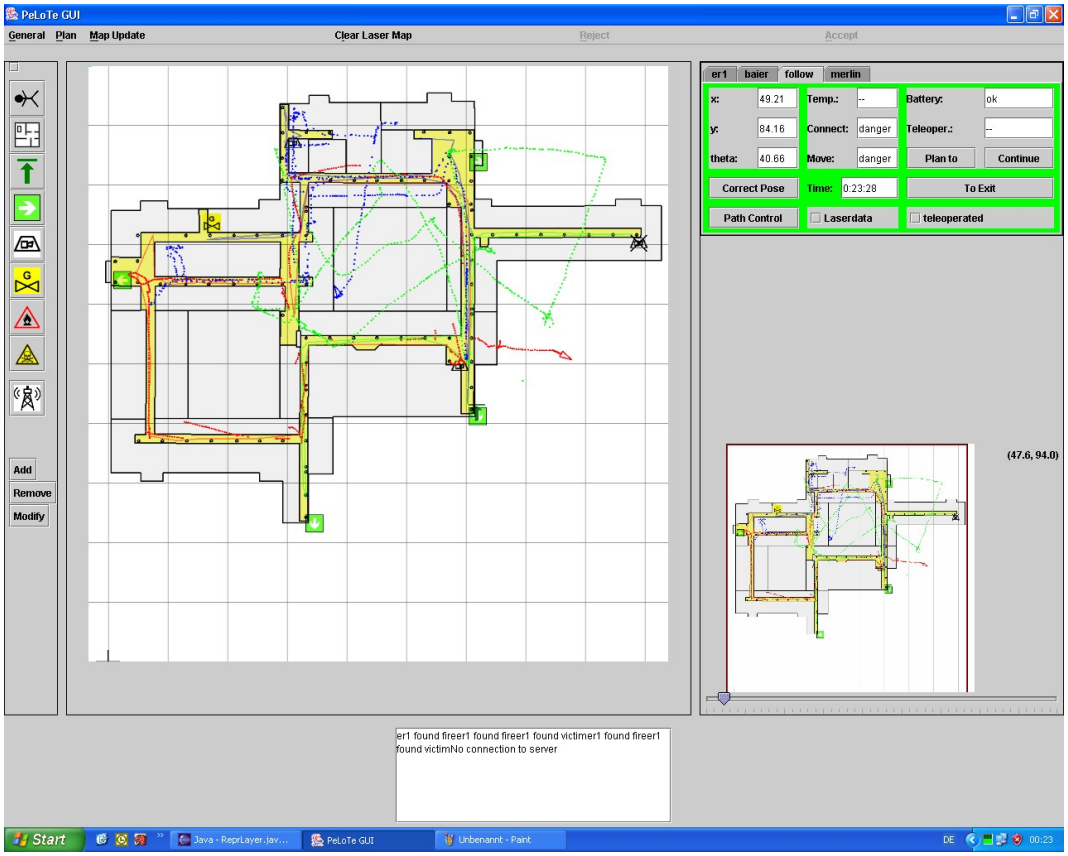


Figure 5.17: An example of the situational view in the semi-final experiment

correction several times during a mission. Since the position correction had to be done frequently some people used it wrongly and changed an approximately correct position to a wrong one. Incorrect localisation information had a significant influence on the performance. The incorrect location created confusion, and the information to the operator came from different coordinate systems. Figure 5.17 shows a situational view of one of the runs in the semi-finals. In the figure, all the entities (2 robots and a human one) have incorrect locations. For the operator and human entity it is impossible to keep track of the events. Additionally, the false location information actually causes confusion when trying to remember the events (as is visible from the memory test).

Strictly speaking, it is impossible to be convinced by the data that the location information is the only cause of the bad results in the semi-finals. The users were not familiar with the system and therefore the malfunction caused confusion, which took the focus off the mission. However, the evidence favours the suggestion that correct position information is one of the keys when creating situational awareness. During the final experiment the PeNa system was working well. Table 5.17 summarises the final experiment from the PeNa point of view. Manual position correction was used only a few times. Figure 5.18 shows the path and integrated laser data of the winning team. The user had a precise location throughout the mission.

Table 5.17: PeNa results of six PeLoTe teams in the final experiments

Team	Path length	Total time	Position corrections
2	313m	1440s	2
4	280m	1320s	0
6	264m	1500s	0
8	246m	1380s	0
10	320m	1740s	3
12	204m	1680s	1

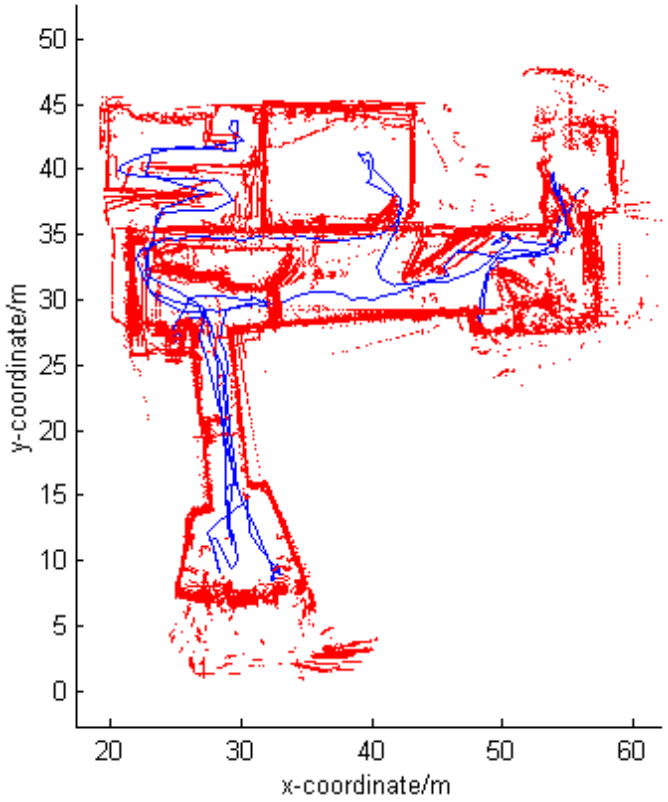


Figure 5.18: The localisation data of the second PeLoTe team during the final experiment

Chapter 6

Conclusion

The main topic of this thesis was the sensor-based localisation of a human being. Sensor-based localisation promises location information that is not dependent on any external infrastructure. This can extend the applications of personal navigation to areas where GPS or other infrastructure cannot be expected to reach. In this thesis the application area was a human-robot team exploring an unknown building and performing a simulated search and rescue task. Search and rescue missions are potential end applications of personal navigation. The missions are time-critical and for the mission leader the location of the rescue personnel is key information.

On the other hand, human-robot teams are an application area of personal navigation itself. The future vision is of humans and automated work machines (or robots) being able to work together as a team [136]. The ability of humans and robotic workers to work as team-mates requires both parties to have common ground, which makes communication and information sharing between the parties possible. In the PeLoTe tests a common model was found to provide a means to model the situation (and related objects and events) for a human-robot team. The key to maintaining a consistent model was accurate location information.

Within the frames of the thesis it has been shown that sensor-based localisation is a feasible way of localising a human being with a bounded error. The results were obtained indoors, but many of them could also be applied outdoors. The methods in this thesis can be divided into personal dead reckoning methods and map-matching methods. For personal dead reckoning the thesis presents two step length measuring implementations (NUPPU and SiLMU), combined with heading estimation and two laser-based dead reckoning algorithms. For map matching the thesis presented three variations of Monte Carlo Localisation methods: 1) topological MCL; 2) scan-based MCL, and 3) combined topological and scan MCL. The topological method is based on the matching of the movement to a map. The scan-based MCL algorithm is often used in robotic applications. Its use in this application domain is new. Finally, the combined algorithm was demonstrated to be the best in most of the cases.

The method that was developed was implemented in a demonstrator called PeNa. PeNa was used to provide the location, in real time, of a "fire-fighter" in a simulated

search and rescue mission. PeNa was used by several people who were not familiar with the system and it was used in a different environment than the one it was developed in.

Additionally, a topological MCL algorithm with 3D dead reckoning information was derived using a foot-mounted IMU. As a result it can be stated that the methods can be extended to cover buildings with several floors. The method was also successfully tested with the VTI pedestrian navigation device. The VTI device represents a possible stream of future mobile devices as: 1) the module was body-mounted (with a 3D gyro it could have any orientation), and 2) it uses components that can be found in mobile phones even now. Thus, using the methods presented in this thesis it is possible to construct a mobile device (e.g. a mobile phone) that could provide information on the location of the user practically anywhere.

Bibliography

[1] T.J. Allen. Architecture and communication among product development engineers. *Engineering Management Society, 2000. Proceedings of the 2000 IEEE*, pages 153–158, 2000.

[2] H. Aoki, B. Schiele, and A. Pentland. Realtime personal positioning system for a wearable computer. In *The Third International Symposium on Wearable Computers. Digest of Papers.*, pages 37–43, 1999.

[3] M.S. Arulampalam, S. Maskell, N. Gordon, and T. Clapp. A tutorial on particle filters for online nonlinear/non-gaussian bayesian tracking. *Signal Processing, IEEE Transactions on*, 50(2):174–188, Feb 2002.

[4] E Ayyappa. Normal human locomotion, part 1: Basic concepts and terminology. *Journal of Prosthetics and Orthotics*, 9(1):10–17, 1997.

[5] Tim Bailey. *Mobile Robot Localisation and Mapping in Extensive Outdoor Environments*. PhD thesis, Australian Center for Field Robotics, Department of Aerospace, Mechanical and Mechatronic Engineering, The University of Sydney, August 2002.

[6] D. Barber, S. Leontyev, Bo Sun, L. Davis, D. Nicholson, and J.Y.C. Chen. The mixed-initiative experimental testbed for collaborative human robot interactions. *Collaborative Technologies and Systems, 2008. CTS 2008. International Symposium on*, pages 483–489, May 2008.

[7] S. Beauregard. A helmet-mounted pedestrian dead reckoning system. In Germany: VDE Verlag, editor, *3rd International Forum on Applied Wearable Computing (IFAWC 2006)*, 2006.

[8] S. Beauregard. Omnidirectional pedestrian navigation for first responders. *Positioning, Navigation and Communication, 2007. WPNC '07. 4th Workshop on*, pages 33–36, 2007.

[9] P.J. Besl and H.D. McKay. A method for registration of 3-d shapes. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 14(2):239–256, Feb 1992.

[10] P. Biber and W. Strasser. The normal distributions transform: a new approach to laser scan matching. *In proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, 2003.(IROS 2003).*, 3, 2003.

[11] J. Borenstein, B. Everett, , and L Feng. Where am i? sensors and methods for mobile robot positioning. Technical report, University of Michigan, 1996.

[12] E. Bradner and G. Mark. Why distance matters: effects on cooperation, persuasion and deception. *Proceedings of the 2002 ACM conference on Computer supported cooperative work*, pages 226–235, 2002.

[13] J.M. Bradshaw, P.J. Feltovich, M.J. Johnson, L. Bunch, M.R. Breedy, T. Eskridge, H. Jung, J. Lott, and A. Uszok. Coordination in Human-Agent-Robot Teamwork. *Collaborative Technologies and Systems, 2008. CTS 2008. International Symposium on*, pages 467–476, 2008.

[14] M. Brady, H. Durrant-Whyte, H. Hu, J. Leonard, P. Probert, and B.S.Y. Rao. Sensor-based control of agvs. *Computing & Control Engineering Journal*, 1(2):64–70, Mar 1990.

[15] D.J. Bruemmer and M.C. Walton. COLLABORATIVE TOOLS FOR MIXED TEAMS OF HUMANS AND ROBOTS. *Multi-Robot Systems: From Swarms to Intelligent Automata: Proceedings from the 2003 International Workshop on Multi-Robot Systems*, 2003.

[16] W. Burgard, D. Fox, D. Hennig, and T. Schmidt. Estimating the Absolute Position of a Mobile Robot Using Position Probability Grids. *PROCEEDINGS OF THE NATIONAL CONFERENCE ON ARTIFICIAL INTELLIGENCE*, pages 896–901, 1996.

[17] J. Burke and R. Murphy. RSVP: an investigation of remote shared visual presence as common ground for human-robot teams. *ACM SIGCHI/SIGART Human-Robot Interaction*, pages 161–168, 2007.

[18] J.L. Burke, R.R. Murphy, M.D. Coovert, and D.L. Riddle. Moonlight in Miami: Field Study of Human-Robot Interaction in the Context of an Urban Search and Rescue Disaster Response Training Exercise. *Human-Computer Interaction*, 19(1&2):85–116, 2004.

[19] AR Cassandra, LP Kaelbling, and JA Kurien. Acting under uncertainty: discrete Bayesian models for mobile-robotnavigation. *Intelligent Robots and Systems' 96, IROS 96, Proceedings of the 1996 IEEE/RSJ International Conference on*, 2, 1996.

[20] J.A. Castellanos, J. Neira, O. Strauss, and J.D. Tardos. Detecting high level features for mobile robot localization. *Multisensor Fusion and Integration for Intelligent Systems, 1996. IEEE/SICE/RSJ International Conference on*, pages 611–618, Dec 1996.

[21] A. Censi, L. Iocchi, and G. Grisetti. Scan Matching in the Hough Domain. In *Robotics and Automation, 2005. Proceedings of the 2005 IEEE International Conference on*, pages 2739–2744, 2005.

[22] Mike Y. Chen, Timothy Sohn, Dmitri Chmelev, Dirk Haehnel, Jeffrey Hightower, Jeff Hughes, Anthony LaMarca, Fred Potter, Ian Smith, and Alex Varshavsky. Practical metropolitan-scale positioning for gsm phones. *Lecture Notes in Computer Science*, 4206/2006:225–242, 2006.

[23] Seong Yun Cho, Chan Gook Park, and HwaYoung Yim. Sensor fusion and error compensation algorithm for pedestrian navigation system. In *Proceedings of the ICCAS 2003, Gyeongju, Korea*, 2003.

[24] Cisco. Wi-fi based real-time location tracking: Solutions and technology. Online http://www.cisco.com, 26.03. 2008.

[25] T.H. Cormen, C.E Leiserson, R.L. Rivest, and C Stein. *Introduction to Algorithms*. MIT Press, 2001.

[26] L.J. Cox. Blanche: Position estimation for an autonomous robot vehicle. *Intelligent Robots and Systems '89. The Autonomous Mobile Robots and Its Applications. IROS '89. Proceedings., IEEE/RSJ International Workshop on*, pages 432–439, Sep 1989.

[27] C.D. Cramton. The Mutual Knowledge Problem and Its Consequences for Dispersed Collaboration. *Organization Science*, 12(3):346, 2001.

[28] D. Cyganski, J. Duckworth, S. Makarov, W. Michalson, J. Orr, V. Amendolare, J. Coyne, H. Daempfling, D. Hubelbank, H. Parikh, and B. Woodacre. Wpi precision personnel locator system. In *Institute of Navigation, National Technical Meeting, San Diego, CA*, 2007.

[29] G.N. Desouza and A.C. Kak. Vision for mobile robot navigation: a survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(2):237–267, 2002.

[30] A. Diosi and L. Kleeman. Laser scan matching in polar coordinates with application to SLAM. *In proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, 2005.(IROS 2005). 2005*, pages 3317–3322, 2005.

[31] M. Dissanayake, P. Newman, S. Clark, HF Durrant-Whyte, and M. Csorba. A solution to the simultaneous localization and map building (slam) problem. *Robotics and Automation, IEEE Transactions on*, 17(3):229–241, 2001.

[32] F. Driewer, H. Baier, and K. Schilling. Robot/Human Interfaces For Rescue Teams. *Proceedings of IFAC Symposium on Telematics Applications in Automation and Robotics, Helsinki*, 2004.

[33] F. Driewer, H. Baier, and K. Schilling. Robot-human rescue teams: a user requirements analysis. *Advanced Robotics*, 19(8):819–838, 2005.

[34] F. Driewer, M. Sauer, and K. Schilling. Design and Evaluation of a Teleoperation Interface for Heterogeneous Human-Robot Teams. *Proc. of the 10th IFAC/IFIP/IFORS/IEA Symposium on Analysis, Design, and Evaluation of Human-Machine Systems*, 2007.

[35] F. Driewer, K. Schilling, and H. Baier. Human-computer interaction in the PeLoTe rescue system. *Safety, Security and Rescue Robotics, Workshop, 2005 IEEE International*, pages 139–144, 2005.

[36] Frauke Driewer. *Teleoperation Interfaces in Human-Robot Teams*. PhD thesis, Julius-Maximilians University of Wuerzburg, Institute of Coputer Science, 2009.

[37] Frauke Driewer, Herbert Baier, Klaus Schilling, Jiri Pavlicek, Libor Preucil, Niramon Ruangpayoongsak, Hubert Roth, Jari Saarinen, Jussi Suomela, Aarne Halme, and Miroslav Kulich. Hybrid telematic teams for search and rescue operations. In *IEEE International Workshop on Safety, Security, and Rescue Robotics (SSRR 2004)*, Bonn, Germany, May 24-26, 2004.

[38] H. Durrant-Whyte, D. Pagac, B. Rogers, M. Stevens, and G. Nelmes. Field and service applications - an autonomous straddle carrier for movement of shipping containers - from research to operational autonomous systems. *Robotics & Automation Magazine, IEEE*, 14(3):14–23, Sept. 2007.

[39] H.F. Durrant-Whyte and J.J. Leonard. Navigation by correlating geometric sensor data. *Intelligent Robots and Systems '89. The Autonomous Mobile Robots and Its Applications. IROS '89. Proceedings., IEEE/RSJ International Workshop on*, pages 440–447, Sep 1989.

[40] F.T. Durso and A. Sethumadhavan. Situation Awareness: Understanding Dynamic Environments. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, 50(3):442–448, 2008.

[41] Ekahau. *Ekahau Positioning Engine*. http://www.ekahau.com/ Last visited 12.03.2008.

[42] A. Elfes. Occupancy grids: A stochastic spatial representation for active robot perception. *Proceedings of the Sixth Conference on Uncertainty in AI*, pages 60–70, 1990.

[43] Mikko Elomaa. Ultrasonic sensor systems for human localisation. Master's thesis, Helsinki University of Technology, Automation Technology Laboratory, 2004.

[44] M. Endsley and W.M. Jones. Situation Awareness Information Dominance & Information Warfare., 1997.

[45] M.R. Endsley. Theoretical underpinnings of situation awareness: A critical review. *Situation Awareness analysis and measurement*, 2000.

[46] Frederic Evennou and Francois Marx. Advanced integration of wifi and inertial navigation systems for indoor mobile positioning. *EURASIP J. Appl. Signal Process.*, 2006(1):164–164.

[47] Lei Fang, P.J. Antsaklis, L.A. Montestruque, M.B. McMickell, M. Lemmon, Yashan Sun, Hui Fang, I. Koutroulis, M. Haenggi, Min Xie, and Xiaojuan Xie. Design of a wireless assisted pedestrian dead reckoning system - the navmote experience. *Instrumentation and Measurement, IEEE Transactions on*, 54(6):2342–2358, Dec. 2005.

[48] P. Felzenszwalb and D. Huttenlocher. Distance Transforms of Sampled Functions. *Cornell Computing and Information Science Technical Report TR2004-1963*, 2004.

[49] T. Fong, C. Kunz, L.M. Hiatt, and M. Bugajska. The human-robot interaction operating system. *Proceedings of the 1st ACM SIGCHI/SIGART conference on Human-robot interaction*, pages 41–48, 2006.

[50] T. Fong, I. Nourbakhsh, C. Kunz, L. Fluckiger, J. Schreiner, R. Ambrose, R. Burridge, R. Simmons, L.M. Hiatt, A. Schultz, et al. The peer-to-peer human-robot interaction project. *Space*, 6750, 2005.

[51] T. Fong, J. Scholtz, J.A. Shah, L. Fluckiger, C. Kunz, D. Lees, J. Schreiner, M. Siegel, L.M. Hiatt, I. Nourbakhsh, et al. A Preliminary Study of Peer-to-Peer Human-Robot Interaction. *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics (SMC)*, 4, 2006.

[52] P. Forsman. Three-Dimensional Localization and Mapping of Static Environments by Means of Mobile Perception. *Helsinki University of technology, Doctor Thesis*, 2001.

[53] D. Fox, W. Burgard, F. Dellaert, and S. Thrun. Monte Carlo Localization: Efficient Position Estimation for Mobile Robots. *PROCEEDINGS OF THE NATIONAL CONFERENCE ON ARTIFICIAL INTELLIGENCE*, pages 343–349, 1999.

[54] E. Foxlin. Pedestrian tracking with shoe-mounted inertial sensors. *Computer Graphics and Applications, IEEE*, 25(6):38–46, Nov.-Dec. 2005.

[55] B. Gerkey, R.T. Vaughan, and A. Howard. The player/stage project: Tools for multi-robot and distributed sensor systems. *Proceedings of the 11th International Conference on Advanced Robotics*, pages 317–323, 2003.

[56] M.A. Goodrich and A.C. Schultz. Human-Robot Interaction: A Survey. *Foundations and Trends® in Human-Computer Interaction*, 1(3):203–275, 2007.

[57] N. J. Gordon, D. J. Salmond, and A. F. M. Smith. Novel approach to nonlinear/non-gaussian bayesian state estimation. *Radar and Signal Processing, IEE Proceedings F*, 140(2):107–113, 1993.

[58] S.A. Green, M. Billinghurst, X.Q. Chen, and J.G. Chase. Human-Robot Collaboration: A Literature Review and Augmented Reality Approach in Design. *International Journal of Advanced Robotic Systems*, 5(1), 2008.

[59] J. Guivant, E. Nebot, and S. Baiker. Localization and map building using laser range sensors in outdoor applications. *Journal of Robotic Systems*, 17(10):565–583, 2000.

[60] J.-S. Gutmann and K. Konolige. Incremental mapping of large cyclic environments. *Computational Intelligence in Robotics and Automation, 1999. CIRA '99. Proceedings. 1999 IEEE International Symposium on*, pages 318–325, 1999.

[61] J. S Gutmann and C. Schlegel. Amos: comparison of scan matching approaches for self-localization in indoor environments. In *Proceedings of the First Euromicro Workshop on Advanced Mobile Robot*, pages 61–67, 1996.

[62] J. S Gutmann, Thilo Weigel, and B. Nebel. Fast, accurate, and robust self-localization in polygonal environments. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 3, pages 1412–1419 vol.3, 1999.

[63] J.S. Gutmann, W. Burgard, D. Fox, and K. Konolige. An experimental comparison of localization methods. *In proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems, 2003.(IROS 2003).*, 2, 1998.

[64] J.M. Hammersley and D.C. Handscomb. *Monte Carlo methods.* Methuen's Monographs on applied probability and statistics, 1964.

[65] Seppo Heikkila. Development of laser based localisation methods for personal navigation system-pena. Master's thesis, Helsinki University Of Technology, Automation Technology Laboratory, 2005.

[66] Edith Herrera, Ricardo Quiros, and Hannes Kaufmann. Analysis of a kalman approach for a pedestrian positioning system in indoor environments. *Euro-Par 2007 Parallel Processing*, 2007.

[67] Y. Inagaki, H. Sugie, H. Aisu, S. Ono, and T. Unemi. Behavior-based intention inference for intelligent robots cooperating with human. *Fuzzy Systems, 1995. International Joint Conference of the Fourth IEEE International Conference on Fuzzy Systems and The Second International Fuzzy Engineering Symposium., Proceedings of 1995 IEEE International Conference on*, 3:1695–1700 vol.3, Mar 1995.

[68] Multispectral Solutions Inc. Online http://www.multispectral.com/, 27.03. 2008.

[69] R. Jirawimut, M. A. Shah, P. Ptasinski, F. Cecelja, and W. Balachandran. Integrated dgps and dead reckoning for a pedestrian navigation system in signal blocked environments. In *ION GPS 2000, 13th International Technical Meeting of the Satellite Division of the Institute of Navigation, Session D4 - GPS - Reference and Information Fusion*, September 19-22, 2000.

[70] H. Jones and P. Hinds. Extreme work teams: using SWAT teams as a model for coordinating distributed robots. *Proceedings of the 2002 ACM conference on Computer supported cooperative work*, pages 372–381, 2002.

[71] D.B. Kaber and M.R. Endsley. Team Situation Awareness for Process Control Safety and Performance. *Process Safety Progress (Vol. 17, No. 1)*, pages 43 – 48, 1998.

[72] J. Kappi, J. Saarinen, and J. Syrjarinne. Mems-imu based pedestrian navigator for handheld devices. *ION GPS, Salt Lake City, UT*, 2001.

[73] Atsuo Kato and Zhiwei Luo. A measurement of stride during human walking. In *SICE*, pages 1105–1108, 1987.

[74] Tobias Kaupp, Bertrand Douillard, Fabio Ramos, Alexei Makarenko, and Ben Upcroft. Shared environment representation for a human-robot team performing information fusion. *J. Field Robot.*, 24(11-12):911–942, 2007.

[75] H. Kawata, A. Ohya, S. Yuta, W. Santosh, and T. Mori. Development of ultra-small lightweight optical range sensor system. *Intelligent Robots and Systems, 2005.(IROS 2005). 2005 IEEE/RSJ International Conference on*, pages 1078–1083, 2005.

[76] W.G. Kennedy, M.D. Bugajska, M. Marge, W. Adams, B.R. Fransen, D. Perzanowski, A.C. Schultz, and J.G. Trafton. Spatial Representation and Reasoning for Human-Robot Collaboration. *PROCEEDINGS OF THE NATIONAL CONFERENCE ON ARTIFICIAL INTELLIGENCE*, 22(2):1554, 2007.

[77] J. W. Kim, H. J. Jang, D.-H. Hwang, and C. Park. A step, stride and heading determination for the pedestrian navigation system. *Journal of Global Positioning Systems*, 3, 2004.

[78] Kurt Konolige and Ken Chou. Markov localization using correlation. In *IJCAI '99: Proceedings of the Sixteenth International Joint Conference on Artificial Intelligence*, pages 1154–1159. Morgan Kaufmann Publishers Inc., 1999.

[79] A. Kotanen, M. Hannikainen, H. Leppakoski, and T.D. Hamalainen. Positioning with ieee 802.11b wireless lan. *Personal, Indoor and Mobile Radio Communications, 2003. PIMRC 2003. 14th IEEE Proceedings on*, 3:2218–2222 vol.3, 7-10 Sept. 2003.

[80] Masakatsu Kourogi and Takeshi Kurata. Personal positioning based on walking locomotion analysis with self-contained sensors and a wearable camera. In *ISMAR '03: Proceedings of the The 2nd IEEE and ACM International Symposium on Mixed and Augmented Reality*, page 103, Washington, DC, USA, 2003. IEEE Computer Society.

[81] Masakatsu Kourogi, Nobuchika Sakata, Takashi Okuma, and Takeshi Kurata. Indoor/outdoor pedestrian navigation with an embedded gps/rfid/self-contained sensor system. *Advances in Artificial Reality and Tele-Existence*, pages 1310–1321, 2006.

[82] M. Kulich, J. Faigl, and L. Preucil. Cooperative planning for heterogeneous teams in rescue operations. *Safety, Security and Rescue Robotics, Workshop, 2005 IEEE International*, pages 18–23, 2005.

[83] Miroslav Kulich, Jan Kout, Libor Preucil, Roman Mazl, Jan Chudoba, Jari Saarinen, Jussi Suomela, Aarne Halme, Frauke Driewer, Herbert Baier, Klaus Schilling, Niramon Ruangpayoongsak, and Hubert Roth. Pelote - a heterogeneous telematic system for cooperative search and rescue missions. In *Urban search and rescue: from Robocup to real world applications, in conjunction with the 2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2004)*, Sendai International Center, Sendai, Japan, September 28, 2004.

[84] Q. Ladetto. On foot navigation : continuous step calibration using both complementary recursive prediction and adaptive kalman filtering. *ION GPS 2000*, 2000.

[85] Anthony LaMarca, Yatin Chawathe, Sunny Consolvo, Jeffrey Hightower, Ian Smith, James Scott, Timothy Sohn, James Howard, Jeff Hughes, Fred Potter, Jason Tabert, Pauline Powledge, Gaetano Borriello, and Bill Schilit. Place lab: Device positioning using radio beacons in the wild. *Pervasive Computing*, pages 116–133, 2005.

[86] W. Lee, H. Ryu, G. Yang, H. Kim, Y. Park, and S. Bang. Design guidelines for map-based human–robot interfaces: A colocated workspace perspective. *International Journal of Industrial Ergonomics*, 37(7):589–604, 2007.

[87] J.J. Leonard and H.F. Durrant-Whyte. Mobile robot localization by tracking geometric beacons. *Robotics and Automation, IEEE Transactions on*, 7(3):376–382, Jun 1991.

[88] H. Leppakoski, J. Kappi, J. Syrjarinne, and J. Takala. Error analysis of step length estimation in pedestrian dead reckoning. In *ION GPS 2002. Portland, OR: The Institute of Navigation*, September 24 -27 2002.

[89] K. Lingemann, A. Nuchter, J. Hertzberg, and H. Surmann. High-speed laser localization for mobile robots. *Robotics and Autonomous Systems*, 51(4):275–296, 2005.

[90] Feng Lu and E. E. Milios. Robot pose estimation in unknown environments by matching 2d range scans. In *Proceedings of Computer Society Conference on Computer Vision and Pattern Recognition*, pages 935–938, 1994.

[91] Bernhard Hofmann-Wellenhof Bernhard Mayerhofer Manfred Wieser and Bettina Pressl. A navigation concept for visually impaired pedestrians in an urban environment. *Vermessung & Geoinformation*, 2:159–165, 2007.

[92] S. May, B. Werner, H. Surmann, and K. Pervolz. 3d time-of-flight cameras for mobile robotics. *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 790–795, 2006.

[93] R. Mazl, M. Kulich, and L. Preucil. Statistical and Feature-Based Methods for Mobile Robot Position Localization. *LECTURE NOTES IN COMPUTER SCIENCE*, pages 517–526, 2001.

[94] R. Mazl, J. Pavlicek, and L. Preucil. Structures for data sharing in hybrid rescue teams. *Safety, Security and Rescue Robotics, Workshop, 2005 IEEE International*, pages 65–70, 2005.

[95] O. Mezentsev, J. Collin, and G. Lachapelle. Pedestrian dead reckoning-a solution to navigation in gps signal degraded areas? *Geomatica*, 59,2:175–182, 2005.

[96] H. Moravec and A. Elfes. High resolution maps from wide angle sonar. In *Proceedings of IEEE International Conference on Robotics and Automation*, volume 2, pages 116–121, 1985.

[97] E. Nebot and H. Durrant-Whyte. Initial Calibration and Alignment of Low-Cost Inertial Navigation Units for Land Vehicle Applications. *Journal of Robotic Systems*, 16(2):81–92, 1999.

[98] J. Neira and JD Tardos. Data association in stochastic mapping using the jointcompatibility test. *Robotics and Automation, IEEE Transactions on*, 17(6):890–897, 2001.

[99] Illah Nourbakhsh. Dervish: an office-navigating robot. *Artificial intelligence and mobile robots: case studies of successful robot systems*, pages 73–90, 1998.

[100] I.R. Nourbakhsh, K. Sycara, M. Koes, M. Yong, M. Lewis, and S. Burion. Human-robot teaming for search and rescue. *Pervasive Computing, IEEE*, 4(1):72–79, Jan.-March 2005.

[101] A. Nuchter, K. Lingemann, and J. Hertzberg. Mapping of rescue environments with kurt3d. *Safety, Security and Rescue Robotics, Workshop, 2005 IEEE International*, pages 158–163, 6-9 June 2005.

[102] International Federation of Robotics. *World Robotics, Executive Summary*. IFR, 2007. Available online http://www.worldrobotics.org/downloads/2007_Executive_Summary.pdf.

[103] Sang Min Oh, S. Tariq, B.N. Walker, and F. Dellaert. Map-based priors for localization. *Intelligent Robots and Systems, 2004. (IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on*, 3:2179–2184 vol.3, Sept.-2 Oct. 2004.

[104] L. Ojeda and J. Borenstein. Non-gps navigation for emergency responders. *International Joint Topical Meeting: "Sharing Solutions for Emergencies and Hazardous Environments", February 12-15, Salt Lake City, Utah, USA.*, 2006.

[105] Lauro Ojeda and Johann Borenstein. Personal dead-reckoning system for gps-denied environments. *Safety, Security and Rescue Robotics, 2007. SSRR 2007. IEEE International Workshop on*, pages 1–6, 27-29 Sept. 2007.

[106] J.M. O'Kane. Global localization using odometry. In *Proceedings of IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, pages 37–42, May 2006.

[107] G.M. Olson and J.S. Olson. Distance Matters. *Human-Computer Interaction*, 15(2/3):139–178, 2000.

[108] V. Otsason, A. Varshavsky, A. LaMarca, and E. de Lara. Accurate gsm indoor localization. *Ubicomp. Volume 3660 of Lecture Notes in Computer Science.*, pages 141–158, 2005.

[109] J. Pavlicek, R. Mazl, L. Preucil, F. Driewer, and K. Schilling. Experiencing New Capabilities of Humans in Hybrid Telematic Teams. In *Biomedical Robotics and Biomechatronics, 2006. BioRob 2006. The First IEEE/RAS-EMBS International Conference on*, pages 353–358, 2006.

[110] M. Popa, J. Ansari, J. Riihijarvi, and P. Mahonen. Combining cricket system and inertial navigation for indoor human tracking. In *Proceedings of IEEE WCNC 2008*, 2008.

[111] Nissanka B. Priyantha, Anit Chakraborty, and Hari Balakrishnan. The cricket location-support system. In *Proceedings of the 6th annual international conference on Mobile computing and networking*, pages 32–43. ACM Press, 2000.

[112] G. Retscher. Location determination in indoor environments for pedestrian navigation. *Position, Location, And Navigation Symposium, 2006 IEEE/ION*, pages 547–555, April 25-27, 2006.

[113] T. Rofer. Using histogram correlation to create consistent laser scan maps. *IEEE/RSJ International Conference on Intelligent Robots and System, 2002*, 1, 2002.

[114] H.J. Roth, R. Schwarte, N. Ruangpayoongsak, J. Kuhle, M. Albrecht, M. Grothof, and H. Hess. 3d vision based on pmd-technology for mobile robots. *Proceedings of SPIE*, 5083, 2003.

[115] J. Saarinen and S. Heikkila. Laser based personal navigation system. *Computational Intelligence in Robotics and Automation, 2005. CIRA 2005. Proceedings. 2005 IEEE International Symposium on*, pages 315–320, 2005.

[116] J. Saarinen, S. Heikkila, M. Elomaa, J. Suomela, and A. Halme. Rescue personnel localization system. *Safety, Security and Rescue Robotics, Workshop, 2005 IEEE International*, pages 104–109, 2005.

[117] Jari Saarinen. D2.2 personal navigation system test report (http://labe.felk.cvut.cz/ pelote/public/peloted2.2final.pdf). Technical report, Helsinki University of Technology, 2004.

[118] Jari Saarinen, Roman Mazl, Petr Ernest, Jussi Suomela, and Libor Preucil. Sensors and methods for human dead reckoning. In *Proceedings of the 8th Conference on Intelligent Autonomous Systems (IAS-8)*, Amsterdam, Netherlands, March 10-13 2004.

[119] Jari Saarinen, Roman Mazl, Miroslav Kulich, Jussi Suomela, Libor Preucil, and Aarne Halme. Methods for personal localisation and mapping. In *Proceedings of the 5th IFAC symposium on Intelligent Autonomous Vehicles, IAV2004*, Lisbon, Portugal, July 5 - 7, 2004.

[120] Jari Saarinen, Jussi Suomela, Aarne Halme, and Jiri Pavlicek. Multientity rescue system. In *IEEE International Workshop on Safety, Security, and Rescue Robotics (SSRR 2004)*, Helsinki, Finland, 21. -23.6.2004.

[121] Jari Saarinen, Jussi Suomela, Seppo Heikkilä, Mikko Elomaa, and Aarne Halme. Personal navigation system. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2004)*, Sendai International Center, Sendai, Japan, September 28 - October 2, 2004.

[122] K. Sagawa, H. Inooka, and Y. Satoh. Non-restricted measurement of walking distance. *Systems, Man, and Cybernetics, 2000 IEEE International Conference on*, 3:1847–1852 vol.3, 2000.

[123] U. Scheunert, H. Cramer, B. Fardi, and G. Wanielik. Multi sensor based tracking of pedestrians: a survey of suitable movement models. *Intelligent Vehicles Symposium, 2004 IEEE*, pages 774–778, 14-17 June 2004.

[124] B. Schiele and J.L. Crowley. A comparison of position estimation techniques using occupancy grids. *Robotics and Automation, 1994. Proceedings., 1994 IEEE International Conference on*, pages 1628–1634 vol.2, May 1994.

[125] K. Schilling and L. Preucil. Tele-presence methods supporting cooperation of robots and humans. *Safety, Security and Rescue Robotics, Workshop, 2005 IEEE International*, pages 217–221, 2005.

[126] A. C. Schultz and W. Adams. Continuous localization using evidence grids. In *Proceedings of IEEE International Conference on Robotics and Automation*, volume 4, pages 2833–2839 vol.4, 1998.

[127] AC Schultz and W. Adams. Continuous localization using evidence grids. *Robotics and Automation, 1998. Proceedings. 1998 IEEE International Conference on*, 4, 1998.

[128] Simon Schuster. Control and user interface for a 3d-laser scanner system. Master's thesis, Helsinki University of Technology, 2007.

[129] J. Selkainaho. ADAPTIVE AUTONOMOUS NAVIGATION OF MOBILE ROBOTS IN UNKNOWN ENVIRONMENTS. *Helsinki University of Technology, Automation Technology Laboratory, Series A: Research reports*, 2002.

[130] J. Selkainaho and P. Forsman. Navigation of an outdoor service robot by matching 2D laser scans. *In Proceedings of IEEE International Symposium on Computational Intelligence in Robotics and Automation, 2005. (CIRA2005)*, pages 353–358, 2005.

[131] Brennan Peter Sellner, Frederik Heger, Laura Hiatt, Reid Simmons, and Sanjiv Singh. Coordinated multi-agent teams and sliding autonomy for large-scale assembly. *Proceedings of the IEEE - Special Issue on Multi-Robot Systems*, 94(7):1425–1444, July 2006.

[132] R. Stirling, J. Collin, K. Fyfe, and G. Lachapelle. An innovative shoe-mounted pedestrian navigation system. In *Proceedings of GNSS 2003, The European navigation Conference Session F3 on Pedestrian Navigation*, pages 22–25, April 2003.

[133] K. Stubbs, D. Wettergreen, and P.J. Hinds. Autonomy and Common Ground in Human-Robot Interaction: A Field Study. *IEEE INTELLIGENT SYSTEMS*, pages 42–50, 2007.

[134] J. Suomela, J. Saarinen, A. Halme, and P. Harmo. Online interactive building of presence. In *Proceedings of The 4th International Conference on Field And Service Robotics*, Springer-Verlag, 2004.

[135] Jussi Suomela, Jari Saarinen, and Aarne Halme. Creating common presence for a multientity rescue team. *in Proc. 16th IFAC World Conference, Prague, July 2005*, page 6, 2005.

[136] Jussi Suomela, Jari Saarinen, Aarne Halme, Matti Vilenius, and Kari Huhtala. Gim - towards the future worksite. *in Proc. 26th IFAC Symposium on Intelligent Autonomous Vehicles, Toulouse September*, 2007.

[137] Juan D Tardos, Jose Neira, Paul M Newman, and John J Leonard. Robust mapping and localization in indoor environments using sonar data. *The International Journal of Robotics Research*, 21(4):311–330, 2002.

[138] Sebastian Thrun, Wolfram Burgard, and Dieter Fox. *Probabilistic Robotics (Intelligent Robotics and Autonomous Agents)*. The MIT Press, 2005.

[139] Sebastian Thrun, Dieter Fox, Wolfram Burgard, and Frank Dallaert. Robust monte carlo localization for mobile robots. *Artif. Intell.*, 128(1-2):99–141, 2001.

[140] Ubisense. Online: http://www.ubisense.net/, 27.03. 2008.

[141] Hui Wang, H. Lenz, A. Szabo, J. Bamberger, and U.D. Hanebeck. Wlan-based pedestrian tracking using particle filters and low-cost mems sensors. *Positioning, Navigation and Communication, 2007. WPNC '07. 4th Workshop on*, pages 1–7, 22-22 March 2007.

[142] Harvey Weinberg. *AN-602 Application Note: Using the ADXL202 in Pedometer and Personal Navigation Applications.* Analog Devices, 2002.

[143] G. Weiss, C. Wetzler, and E. von Puttkamer. Keeping track of position and orientation of moving indoor systems by correlation of range-finder scans. *Intelligent Robots and Systems '94. 'Advanced Robotic Systems and the Real World', IROS '94. Proceedings of the IEEE/RSJ/GI International Conference on*, 1:595–601 vol.1, Sep 1994.

[144] M. Woof. Technology for Underground Loading and Hauling Systems Offers Exciting Prospects. *Engineering and Mining Journal*, 206(3):32–33, 2005.

[145] J. Xavier and U. Nunes. Interfacing with multiple robots using environmental awareness from a multi-modal hri. *The 7th Conference on Mobile Robots and Competitions, Portugal, April*, 2007.

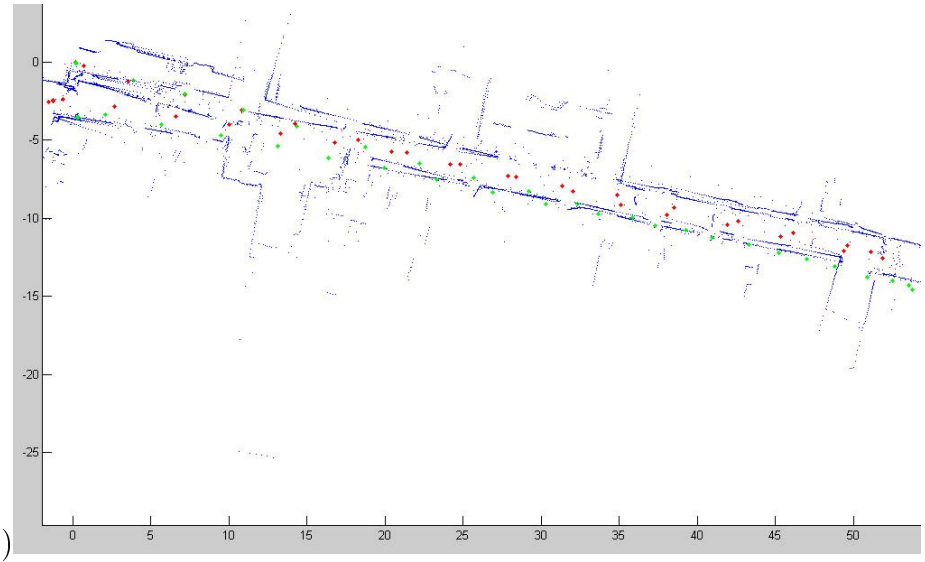
[146] Shun yuan Yeh, Keng hao Chang, Chon in Wu, Hao hua Chu, and Jane Hsu. Geta sandals: a footstep location tracking system. *Personal and Ubiquitous Computing*, 11(6):451–463, August 2007.

Appendices

Appendix A

The Dead Reckoning Results

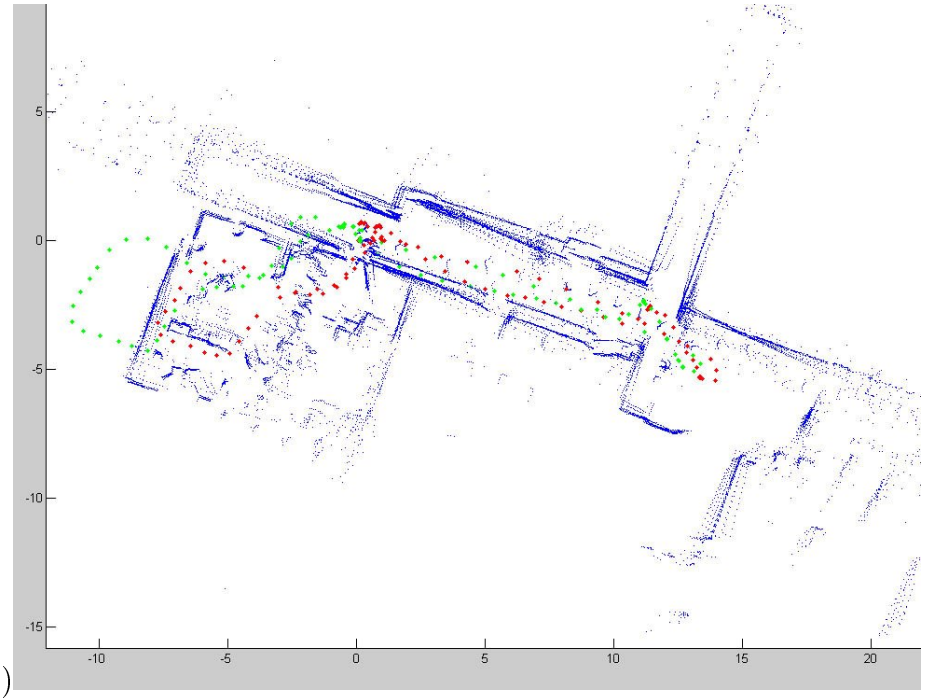
A.1 Case 1: Correlation in the pose space



1)

Figure A.1: Dead reckoning results. set No. 1. 1) reference walk

2)



3)

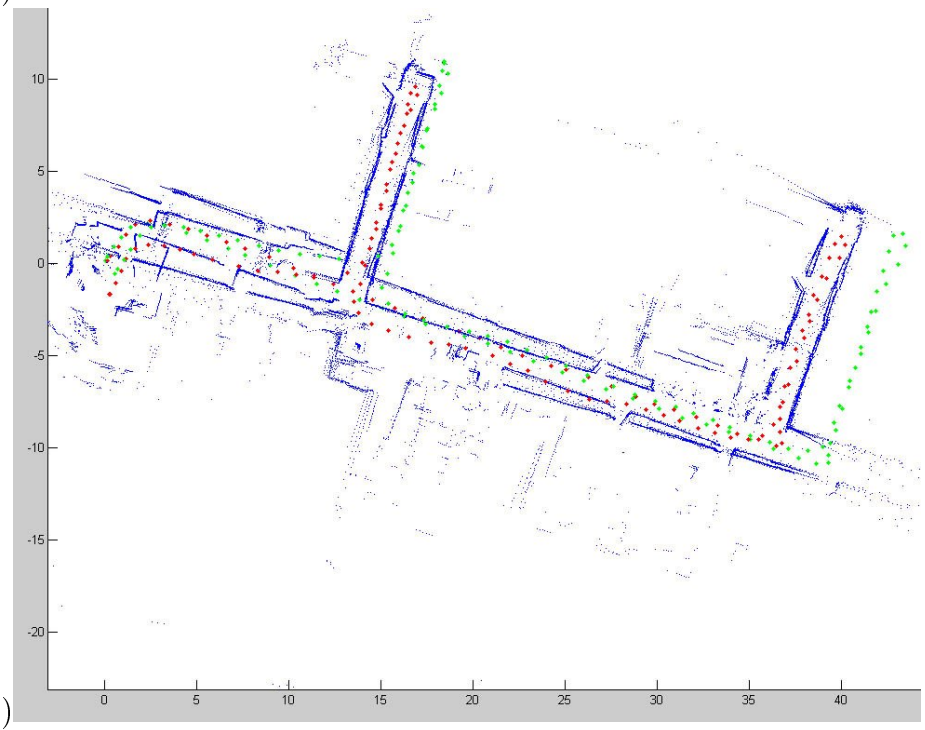


Figure A.2: Dead reckoning results. Sets No. 2 and No. 3. 2) Short corridor walk with one room 3) Short corridor walk

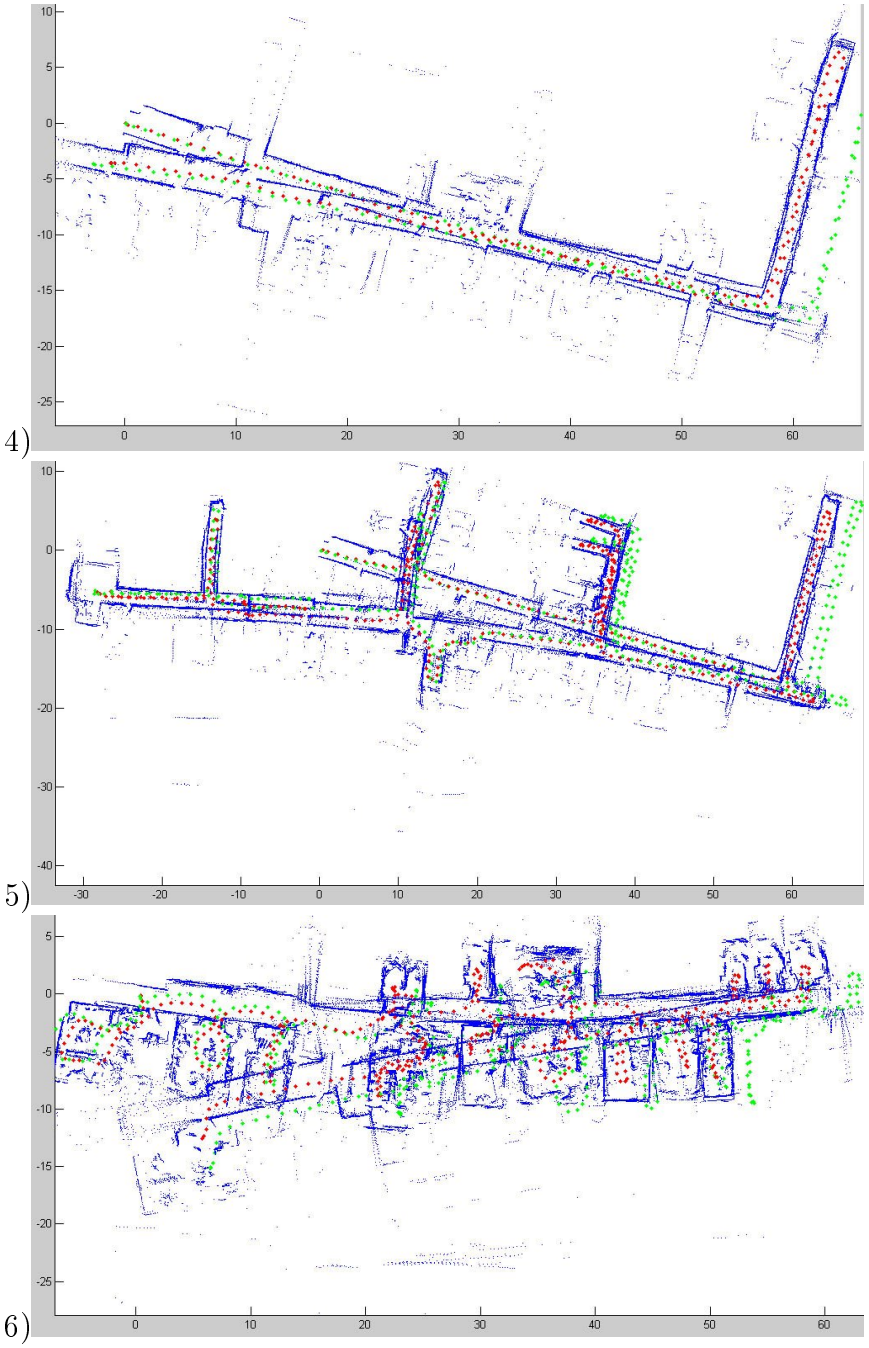
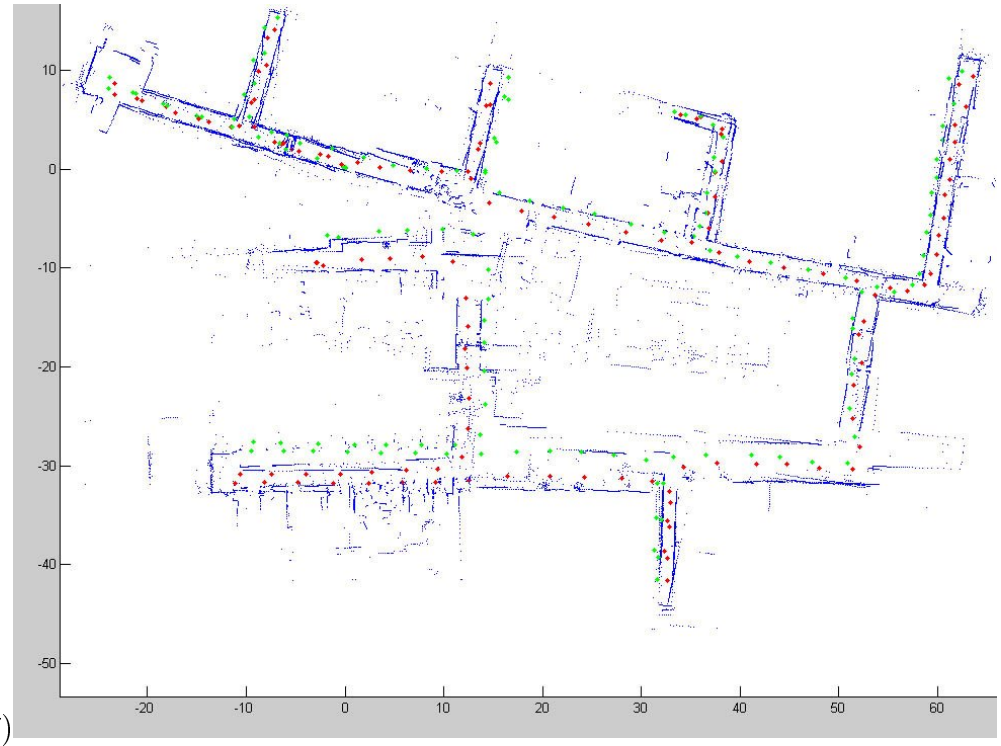


Figure A.3: Dead Reckoning results 4-6. From the top: 4) simple corridor walk 5) long corridor walk 6) room search

7)



8)

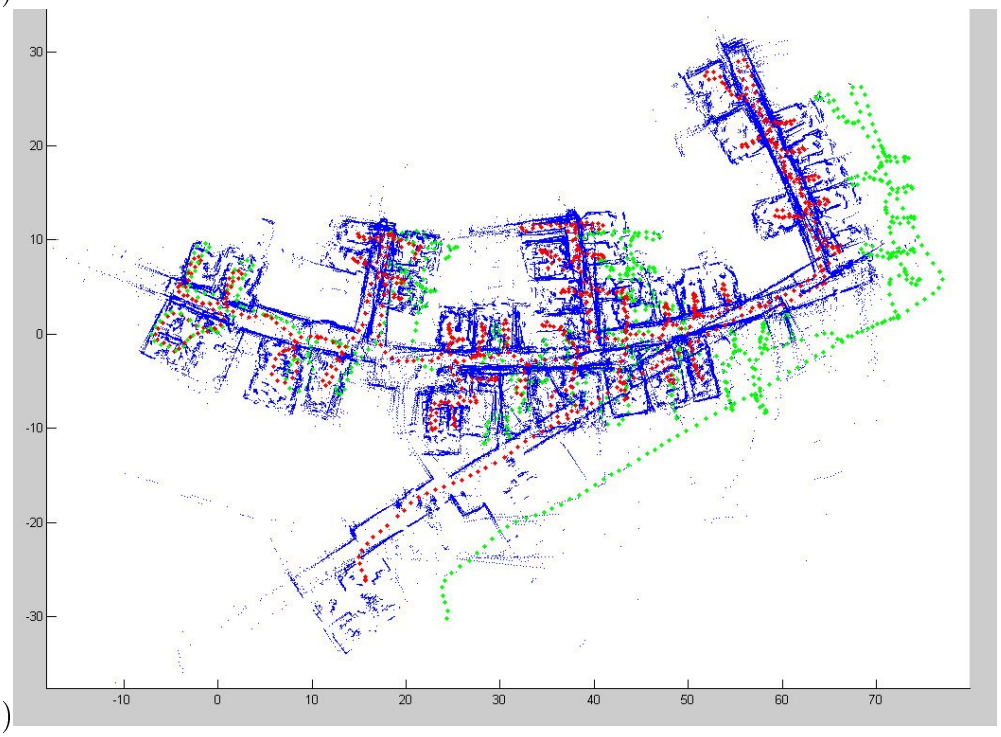


Figure A.4: Dead Reckoning Results 7-8. From the top: 7) Corridor coverage walk 8) Mapping of the whole automation laboratory.

A.2 Case 2: Combined angle and position correlation scan matching

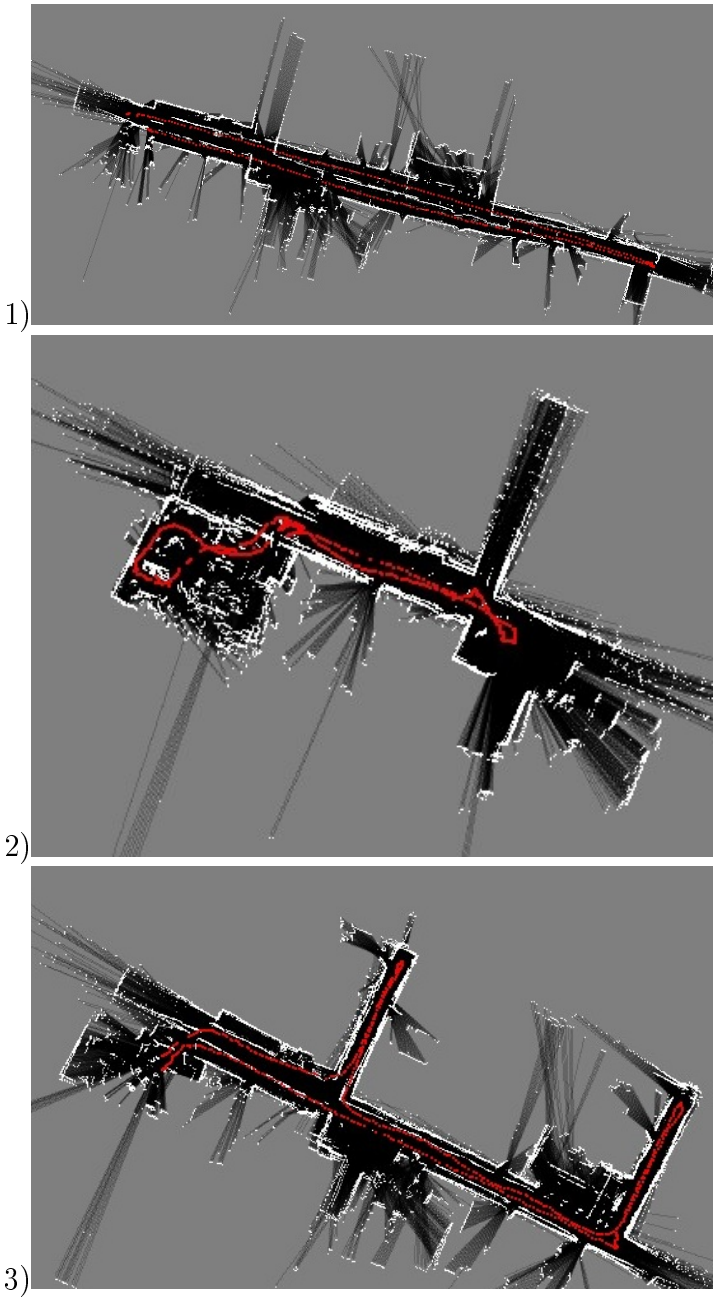


Figure A.5: Dead reckoning results 1-3. From top: 1) reference walk 2) Short corridor walk with one room 3) Short corridor walk

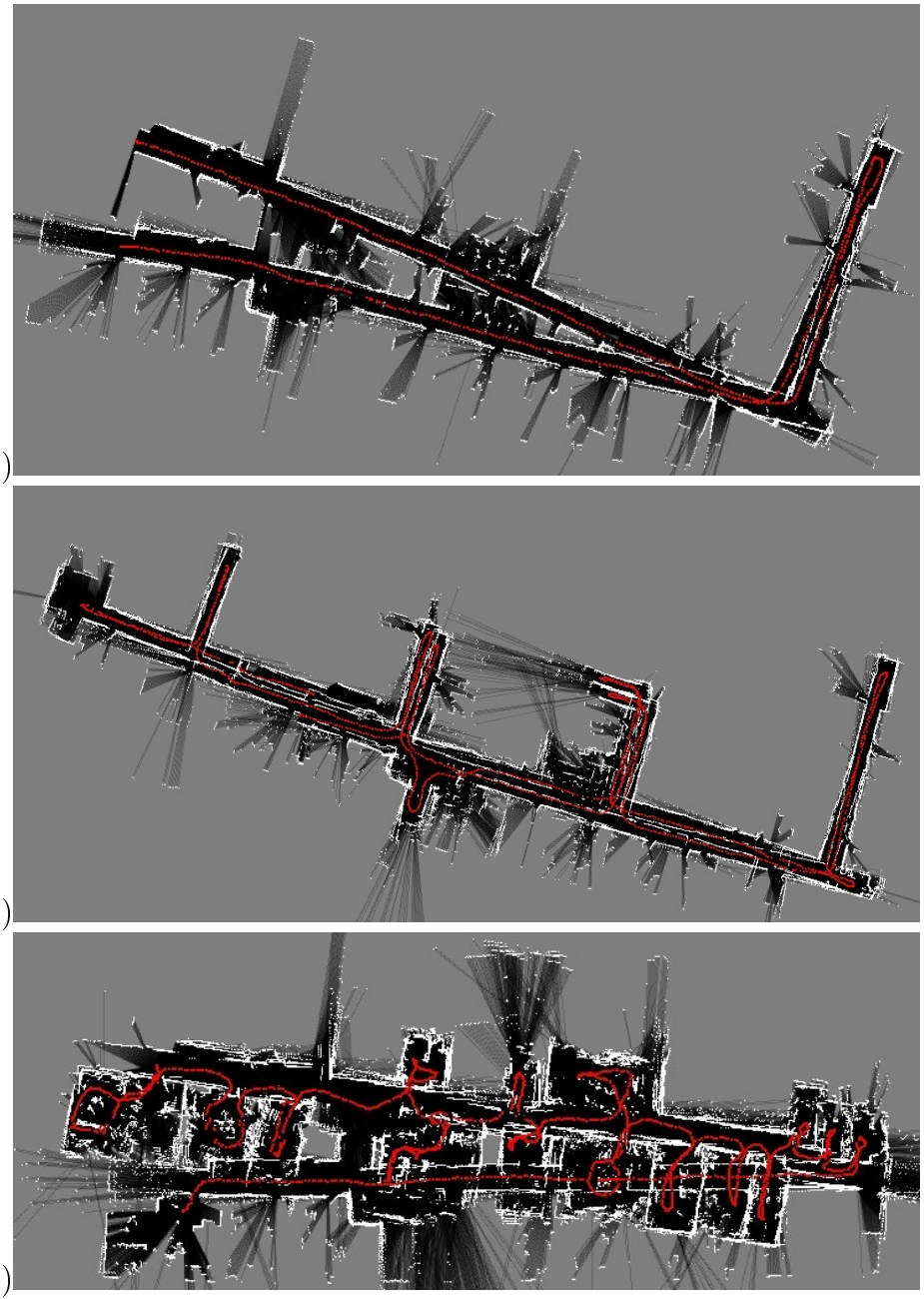


Figure A.6: Dead Reckoning results 4-6. From the top: 4) simple corridor walk 5) long corridor walk 6) room search

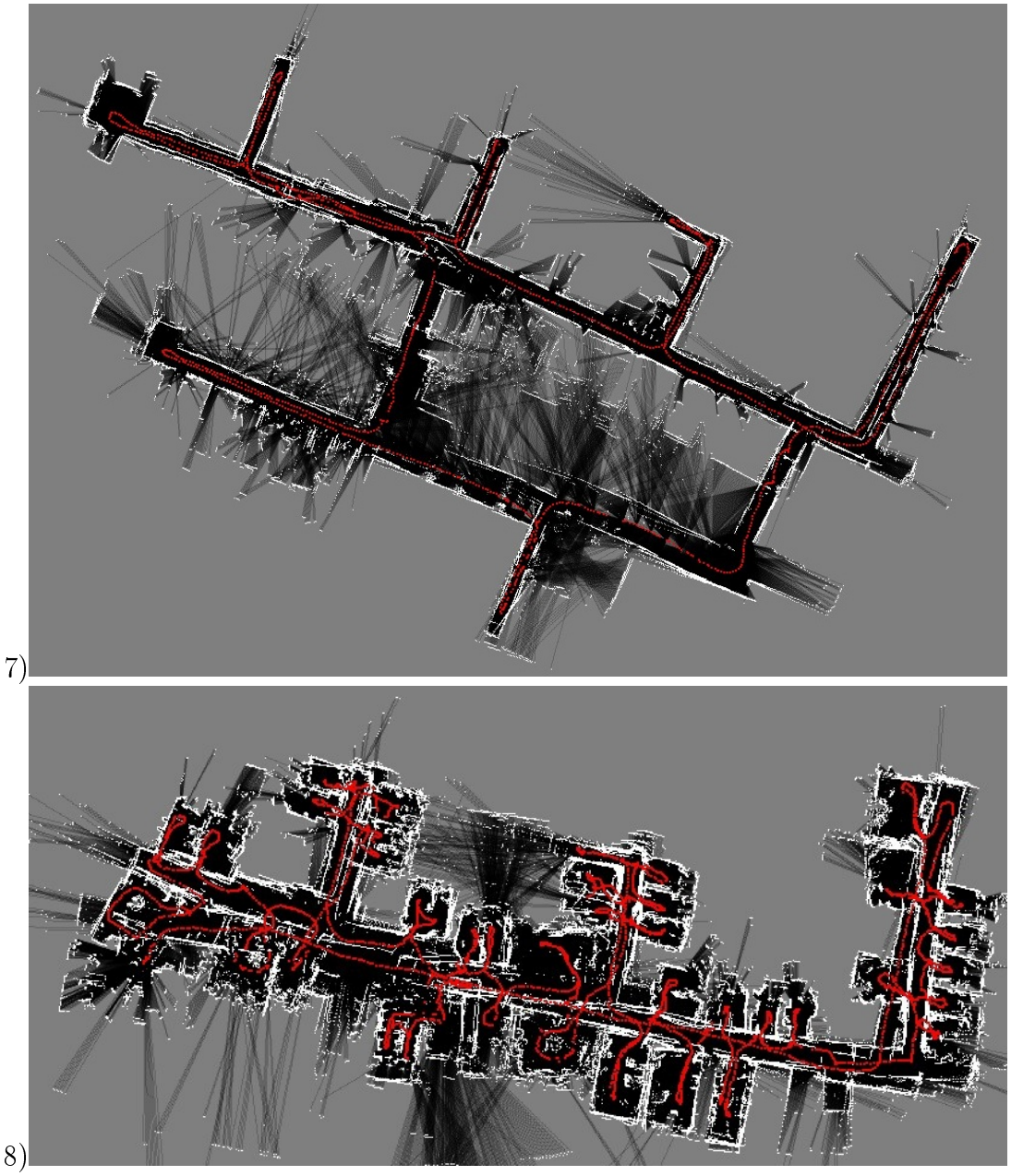


Figure A.7: Dead Reckoning Results 7-8. From the top: 7) Corridor coverage walk 8) Mapping of the whole automation laboratory.

Appendix B

The results of the map based methods

B.1 Tables for all MCL runs

Table B.1: MCL Runs on the set No. 1

Method	Particles	Var(x)	Var(y)	Var(a)	err(x)	err(y)	err(a)
topo	100	2.57	0.58	0.01	-1.29	0.02	0
topo	200	2.62	0.5	0.01	-0.24	0	-0.01
topo	500	2.58	0.62	0.01	-0.65	0	-0.01
topo	1000	2.6	0.83	0.01	-0.28	-0.02	-0.01
topo	5000	2.54	1.75	0.01	-0.44	-0.02	-0.01
topo	10000	2.77	1.89	0	-0.45	-0.02	-0.01
topo	20000	2.66	1.82	0	-0.46	-0.01	-0.01
scan	100	0.52	0.09	0	-0.51	-0.01	-0.01
scan	200	1.55	0.1	0	-1.91	0.08	-0.01
scan	500	0.59	0.06	0	0.05	-0.03	-0.01
scan	1000	0.58	0.08	0	0.25	0.11	-0.02
scan	5000	1.12	0.1	0	0.18	0.08	-0.02
scan	10000	1.32	0.1	0	-0.67	0.06	-0.02
scan	20000	1.08	0.1	0	-0.03	-0.01	-0.01
toposcan	100	0.68	0.06	0	0.1	-0.07	-0.02
toposcan	200	0.66	0.09	0	-0.13	-0.02	-0.01
toposcan	500	0.8	0.1	0	-0.08	-0.05	-0.01
toposcan	1000	0.66	0.12	0	-0.09	-0.06	-0.01
toposcan	5000	0.95	0.2	0.01	0.02	-0.06	-0.01
toposcan	10000	0.96	0.23	0	0.06	-0.06	-0.01
toposcan	20000	1.06	0.3	0	0.01	-0.06	-0.01

Table B.2: MCL Runs on the set No. 2

Method	Particles	Var(x)	Var(y)	Var(a)	err(x)	err(y)	err(a)
topo	100	1.02	0.57	0.02	0.42	-0.04	-0.02
topo	200	0.78	0.44	0.02	0.07	0.03	-0.01
topo	500	0.94	0.59	0.03	0.49	-0.04	-0.02
topo	1000	0.89	0.66	0.03	0.42	-0.05	-0.02
topo	5000	0.97	0.77	0.02	0.45	-0.05	-0.02
topo	10000	1.01	0.87	0.03	0.43	-0.08	-0.02
topo	20000	1	0.95	0.02	0.45	-0.06	-0.02
scan	100	0.15	0.08	0.01	-0.12	-0.04	0
scan	200	0.13	0.05	0	0.11	0.09	-0.01
scan	500	0.18	0.08	0	-0.07	0.02	-0.02
scan	1000	0.18	0.09	0.01	-0.04	0.03	-0.02
scan	5000	0.15	0.11	0.01	-0.11	0	-0.02
scan	10000	0.23	0.13	0.01	-0.11	0	-0.02
scan	20000	0.28	0.22	0	-0.09	0	-0.02
toposcan	100	0.25	0.1	0	0.29	0.01	-0.03
toposcan	200	0.24	0.11	0.01	0.1	0.02	-0.02
toposcan	500	0.29	0.12	0.01	0.1	0.01	-0.02
toposcan	1000	0.28	0.13	0.01	0.08	0	-0.02
toposcan	5000	0.41	0.21	0.01	0.09	0.02	-0.02
toposcan	10000	0.41	0.32	0	0.09	0.01	-0.02
toposcan	20000	0.45	0.32	0	0.1	0.01	-0.02

Table B.3: MCL Runs on the set No. 3

Method	Particles	Var(x)	Var(y)	Var(a)	err(x)	err(y)	err(a)
topo	100	0.59	0.56	0.06	0.13	0.21	0
topo	200	0.59	0.53	0.07	-0.01	0.18	-0.01
topo	500	0.48	0.65	0.09	-0.01	0.1	0.01
topo	1000	0.59	0.69	0.11	0.01	0.05	-0.01
topo	5000	0.66	1.45	0.07	0.03	0.04	-0.01
topo	10000	0.83	1.64	0.09	0.03	0.06	-0.02
topo	20000	0.87	1.81	0.02	0.03	0.06	-0.01
scan	100	0.19	0.31	0.02	0	0.17	-0.01
scan	200	0.25	0.31	0.03	-0.03	0.12	0.04
scan	500	0.27	0.27	0.01	0.01	0.22	0
scan	1000	0.29	0.23	0.03	0.05	0.14	-0.01
scan	5000	0.42	0.23	0	0	0.1	0.02
scan	10000	0.39	0.26	0.03	0.01	0.13	0.01
scan	20000	0.29	0.26	0.02	-0.02	0.11	0
toposcan	100	0.24	0.2	0.06	0.11	0	-0.04
toposcan	200	0.24	0.23	0	0.11	0.01	-0.02
toposcan	500	0.26	0.23	0.08	0.07	0.02	0
toposcan	1000	0.3	0.22	0.04	0.09	0.01	-0.01
toposcan	5000	0.37	0.32	0.05	0.08	0.02	-0.01
toposcan	10000	0.4	0.4	0.04	0.08	0.03	-0.01
toposcan	20000	0.49	0.49	0.05	0.08	0.02	-0.01

Table B.4: MCL Runs on the set No. 4

Method	Particles	Var(x)	Var(y)	Var(a)	err(x)	err(y)	err(a)
topo	100	0.78	0.36	0.01	-0.51	0.01	0.03
topo	200	0.92	0.6	0.01	-0.5	0.09	0.04
topo	500	1.04	0.59	0.02	-0.56	-0.02	0.02
topo	1000	1.11	0.71	0.01	-0.58	-0.03	0.03
topo	5000	1.38	1.34	0.01	-0.61	-0.03	0.02
topo	10000	1.46	1.41	0.05	-0.57	-0.03	0.02
topo	20000	1.28	1.79	0.02	-0.58	-0.03	0.02
scan	100	0.2	0.12	0.03	-0.02	0.08	-0.02
scan	200	0.26	0.13	0.03	-0.12	0.02	0
scan	500	0.14	0.08	0.02	-0.04	0.05	-0.01
scan	1000	0.17	0.09	0.01	-0.05	0.03	-0.03
scan	5000	0.22	0.11	0.02	-0.04	0.06	-0.03
scan	10000	0.23	0.12	0.02	-0.01	0.04	-0.02
scan	20000	0.24	0.14	0.01	-0.04	0.04	-0.02
toposcan	100	0.43	0.15	0	-0.02	0	0
toposcan	200	0.32	0.15	0	-0.13	-0.01	-0.01
toposcan	500	0.29	0.14	0.01	-0.03	-0.02	0
toposcan	1000	0.33	0.16	0.05	-0.05	-0.02	-0.01
toposcan	5000	0.49	0.26	0	-0.04	-0.03	0
toposcan	10000	0.55	0.33	0.04	-0.04	-0.03	-0.01
toposcan	20000	0.59	0.43	0.02	-0.05	-0.02	-0.01

Table B.5: MCL Runs on the set No. 5

Method	Particles	Var(x)	Var(y)	Var(a)	err(x)	err(y)	err(a)
topo	100	0.38	0.46	0.04	-0.07	0.07	0.03
topo	200	0.38	0.48	0.05	-0.02	0.2	0
topo	500	0.41	0.59	0.07	0.01	0.17	0.02
topo	1000	0.46	0.66	0.04	-0.01	0.16	0.03
topo	5000	0.6	0.82	0.05	0	0.16	0.02
topo	10000	0.69	1.05	0.03	0	0.15	0.02
topo	20000	0.7	1.08	0.02	-0.01	0.17	0.02
scan	100	0.22	0.16	0.03	0	0.1	0
scan	200	0.16	0.13	0.03	0	0.12	0.02
scan	500	0.14	0.17	0.02	-0.04	0.05	0.01
scan	1000	0.16	0.14	0.02	0	0.05	0
scan	5000	0.31	0.25	0.01	-0.59	3.42	-0.09
scan	10000	0.25	0.23	0.03	-0.01	0.04	0.01
scan	20000	0.25	0.28	0.02	-0.02	0.06	0
toposcan	100	0.19	0.13	0.02	0.02	0.03	-0.02
toposcan	200	0.23	0.16	0.07	-0.01	0.01	0.01
toposcan	500	0.24	0.16	0.04	0.01	0.03	-0.02
toposcan	1000	0.26	0.17	0.08	0.01	0.03	0
toposcan	5000	0.33	0.25	0.09	0.01	0.04	-0.01
toposcan	10000	0.42	0.32	0.07	0.01	0.04	-0.01
toposcan	20000	0.39	0.39	0.09	0.01	0.03	-0.01

Table B.6: MCL Runs on the set No. 6

Method	Particles	Var(x)	Var(y)	Var(a)	err(x)	err(y)	err(a)
topo	100	0.31	0.49	0.08	-0.28	-0.01	0
topo	200	0.31	0.47	0.05	-0.16	-0.28	-0.02
topo	500	0.33	0.48	0.09	-0.15	-0.22	-0.02
topo	1000	0.39	0.53	0.07	-0.13	-0.23	-0.01
topo	5000	0.5	1	0.06	-0.15	-0.17	-0.02
topo	10000	0.54	1.02	0.08	-0.13	-0.21	-0.01
topo	20000	0.57	1.39	0.08	-0.12	-0.2	-0.01
scan	100	0.09	0.08	0.04	-0.06	-0.09	-0.01
scan	200	0.11	0.08	0.04	-0.06	-0.01	0.02
scan	500	0.14	0.09	0.05	-0.06	-0.02	0
scan	1000	0.17	0.09	0.03	-0.07	-0.02	0.01
scan	5000	0.2	0.14	0.05	-0.06	-0.01	-0.01
scan	10000	0.2	0.15	0.05	-0.06	-0.02	0.01
scan	20000	0.2	0.12	0.05	-0.07	-0.03	0.01
toposcan	100	0.12	0.11	0.06	-0.03	0.02	0.01
toposcan	200	0.13	0.1	0.05	-0.01	-0.04	0
toposcan	500	0.14	0.1	0.08	-0.02	-0.03	0.01
toposcan	1000	0.16	0.12	0.06	-0.02	-0.02	0
toposcan	5000	0.21	0.18	0.05	-0.02	-0.02	0.01
toposcan	10000	0.28	0.22	0.05	-0.01	-0.01	0.01
toposcan	20000	0.3	0.25	0.07	-0.01	-0.02	0.01

Table B.7: MCL Runs on the set No. 7

Method	Particles	Var(x)	Var(y)	Var(a)	err(x)	err(y)	err(a)
topo	100	0.66	0.76	0.05	-0.03	0.28	0.05
topo	200	0.61	0.69	0.02	-0.05	0.07	0.03
topo	500	0.82	0.86	0.01	-0.02	0.04	0.02
topo	1000	0.8	1.01	0.03	-0.04	0.07	0.02
topo	5000	0.91	1.57	0.03	-0.05	0.02	0.02
topo	10000	1.16	1.84	0.02	-0.07	0.03	0.02
topo	20000	1.18	1.9	0.01	-0.05	0.06	0.02
scan	100	0.29	0.23	0.03	-0.14	-0.34	-0.06
scan	200	0.18	0.19	0.05	-0.04	-0.08	-0.08
scan	500	0.31	0.25	0.02	-0.13	-0.18	-0.05
scan	1000	0.31	0.2	0	-0.12	-0.13	-0.06
scan	5000	0.29	0.35	0.01	-0.08	-0.14	-0.07
scan	10000	0.31	0.33	0	-0.09	-0.1	-0.07
scan	20000	0.38	0.51	0.02	-0.08	-0.09	-0.09
toposcan	100	0.39	0.36	0.02	0.02	-0.01	-0.01
toposcan	200	0.43	0.44	0.03	0.05	-0.06	0
toposcan	500	0.46	0.38	0.01	-0.05	-0.02	0
toposcan	1000	0.56	0.4	0.01	0.01	0	-0.01
toposcan	5000	0.63	0.62	0.01	0.03	0	-0.01
toposcan	10000	0.73	0.73	0.02	0.02	0.01	-0.02
toposcan	20000	0.83	1.09	0.01	0.02	0	-0.02

Table B.8: MCL Runs on the set No. 8

Method	Particles	Var(x)	Var(y)	Var(a)	err(x)	err(y)	err(a)
topo	100	0.31	0.26	0.03	-0.11	0.68	-0.08
topo	200	0.26	0.29	0.02	-0.03	-0.16	-0.05
topo	500	0.32	0.36	0.04	0.02	-0.05	-0.02
topo	1000	0.34	0.4	0.04	-0.03	0.02	0
topo	5000	0.53	0.54	0.03	-0.03	0.01	0.01
topo	10000	0.5	0.56	0.03	-0.05	0.02	0
topo	20000	0.58	0.54	0.04	-0.05	0.02	0
scan	100	0.08	0.08	0.01	-1.03	-1.33	-0.27
scan	200	0.08	0.09	0.01	-0.06	0.06	-0.02
scan	500	0.11	0.09	0.02	-0.07	0.06	-0.03
scan	1000	0.09	0.09	0.01	-0.06	0.05	-0.01
scan	5000	0.14	0.11	0	-0.05	0.06	-0.02
scan	10000	0.15	0.13	0	-0.06	0.07	-0.02
scan	20000	0.15	0.13	0.01	-0.07	0.07	-0.02
toposcan	100	0.12	0.1	0.03	-0.31	0.01	-0.05
toposcan	200	0.11	0.09	0.02	-0.02	0	-0.02
toposcan	500	0.12	0.09	0.01	-0.02	0.01	-0.01
toposcan	1000	0.13	0.1	0.01	-0.02	0.01	-0.02
toposcan	5000	0.19	0.16	0.03	-0.02	0.01	-0.02
toposcan	10000	0.21	0.18	0.02	-0.02	0	-0.01
toposcan	20000	0.26	0.22	0.03	-0.02	-0.01	-0.02

B.2 Paths of all MCL runs

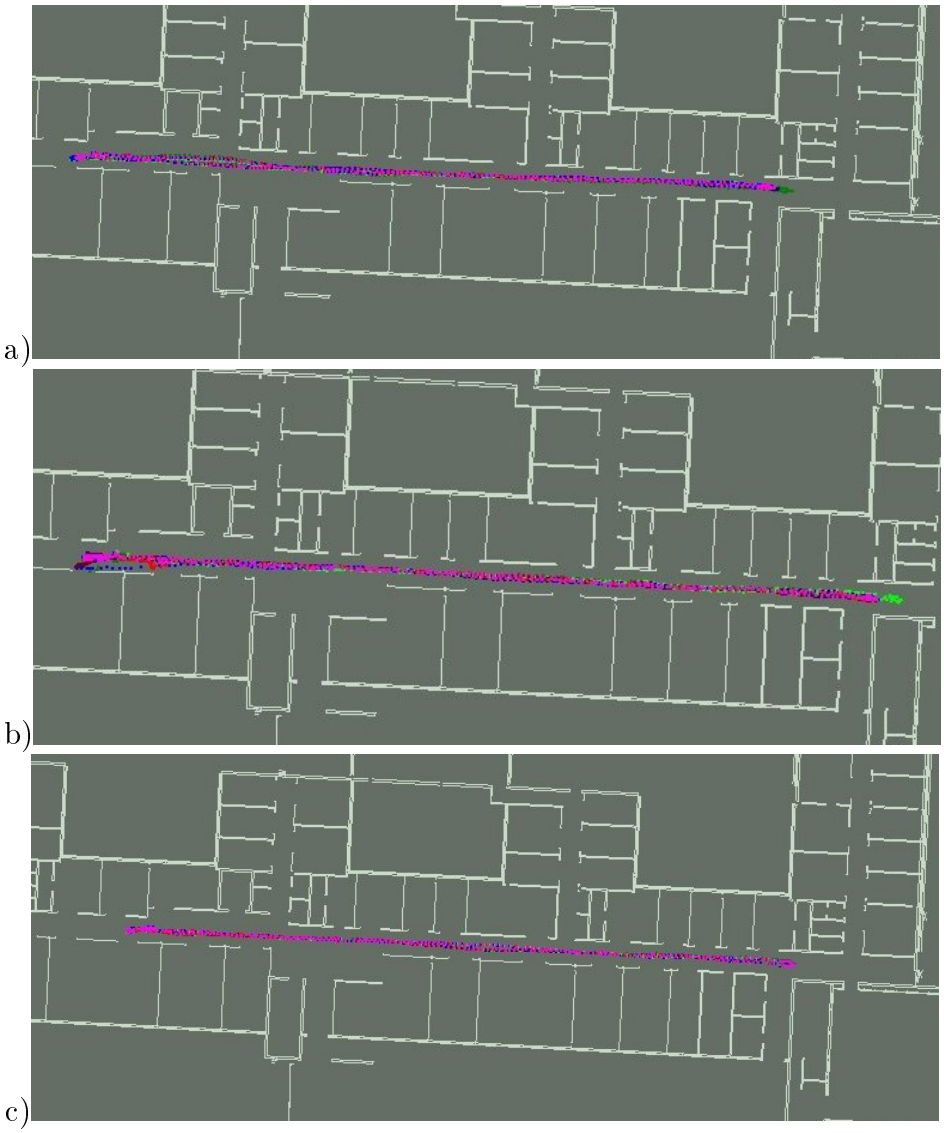


Figure B.1: Paths of the set No. 1. a) is the results obtained with the topo-MCL b) with scan-MCL and c) with topo-scan-MCL.

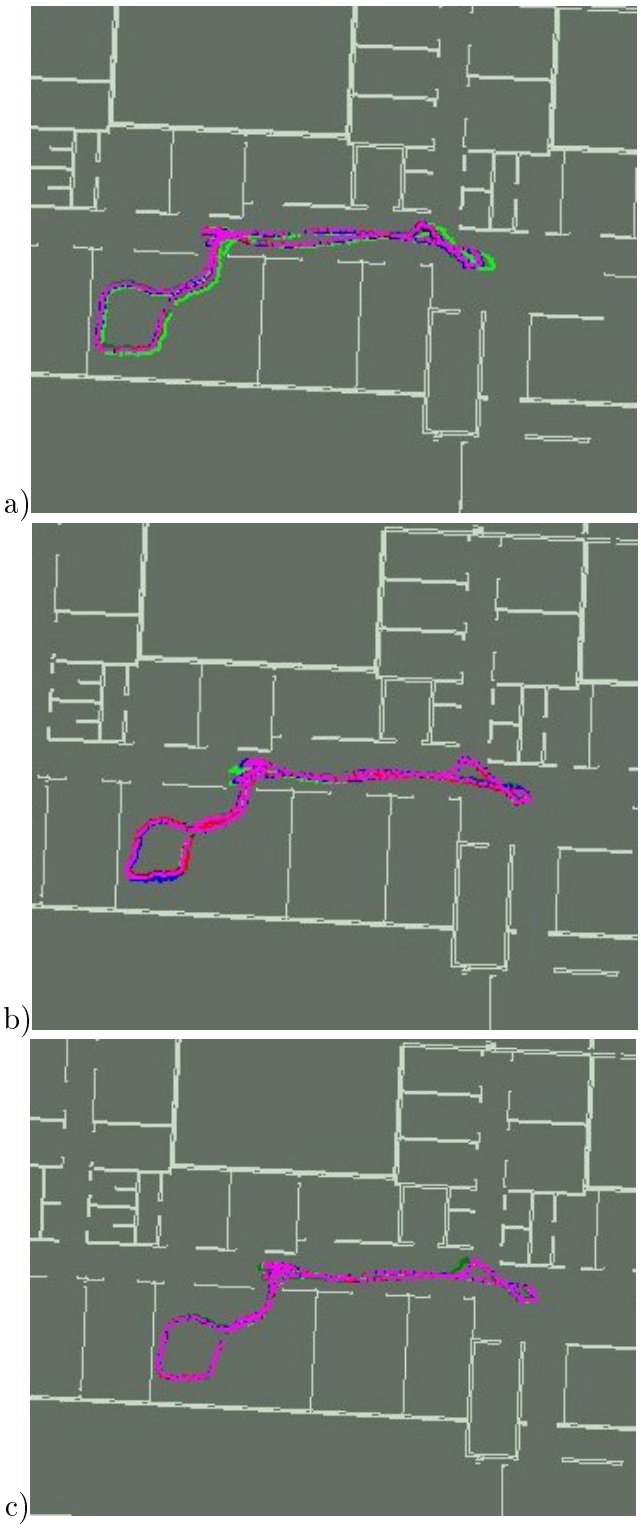


Figure B.2: Paths of the set No. 2. a) is the results obtained with the topo-MCL b) with scan-MCL and c) with topo-scan-MCL.

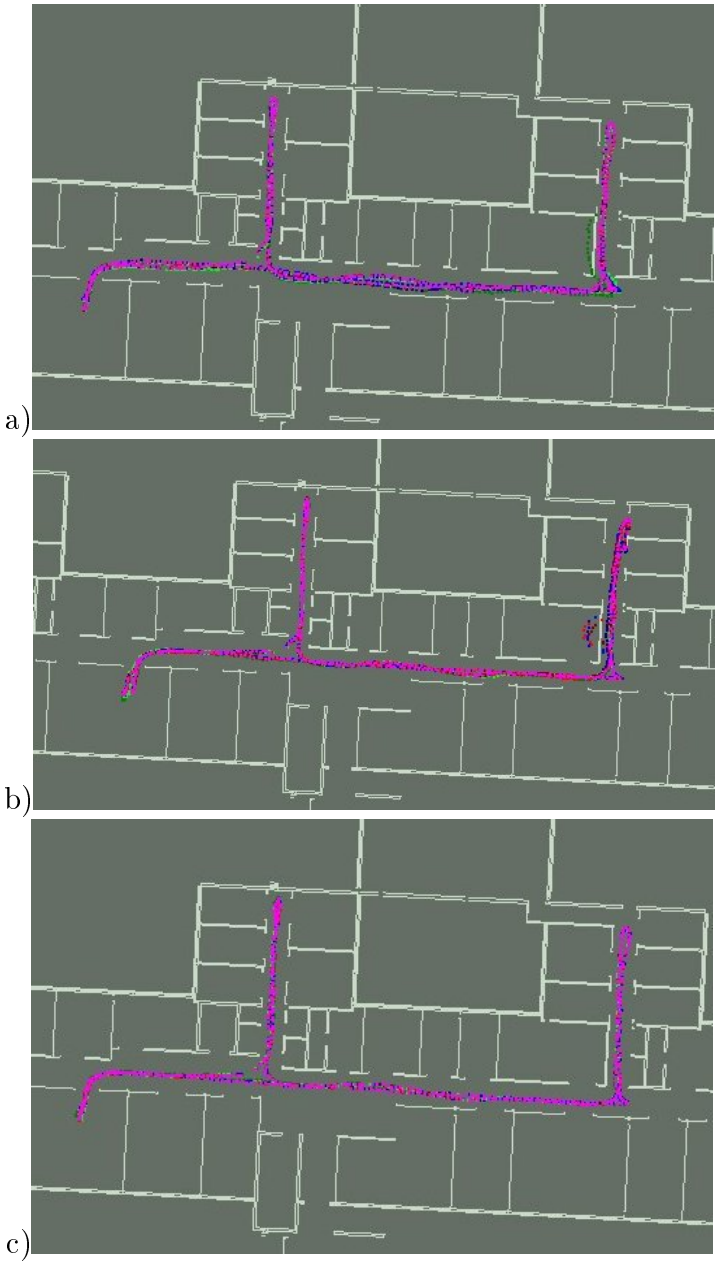


Figure B.3: Paths of the set No. 3. a) is the results obtained with the topo-MCL b) with scan-MCL and c) with topo-scan-MCL.

Figure B.4: Paths of the set No. 4. a) is the results obtained with the topo-MCL b) with scan-MCL and c) with topo-scan-MCL.

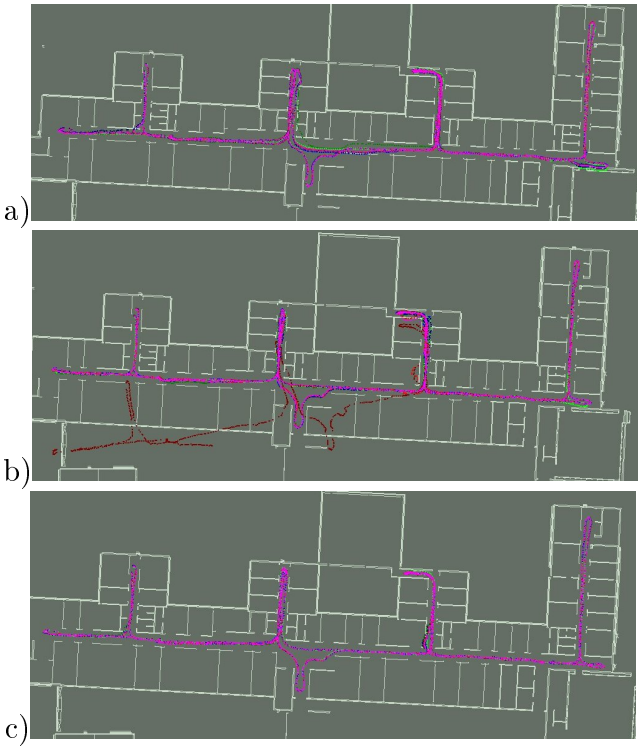


Figure B.5: Paths of the set No. 5. a) is the results obtained with the topo-MCL b) with scan-MCL and c) with topo-scan-MCL.

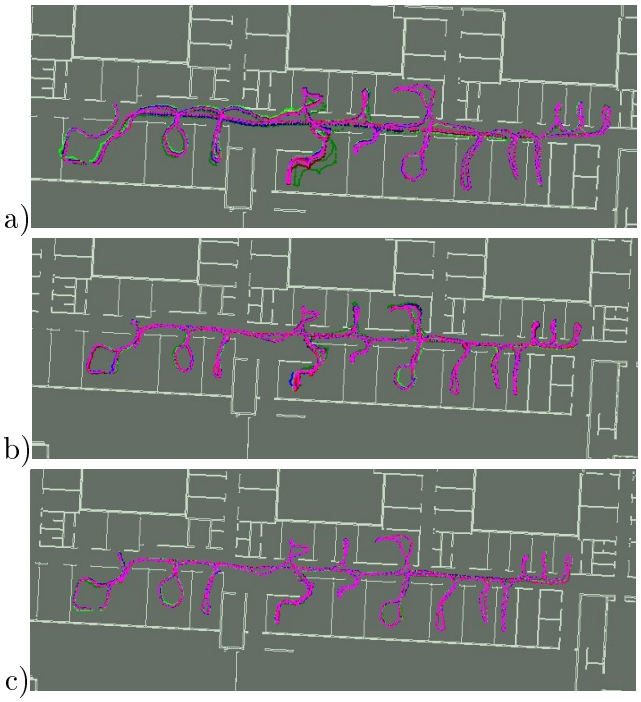


Figure B.6: Paths of the set No. 6. a) is the results obtained with the topo-MCL b) with scan-MCL and c) with topo-scan-MCL.

Figure B.7: Paths of the set No. 7. a) is the results obtained with the topo-MCL b) with scan-MCL and c) with topo-scan-MCL.

Figure B.8: Paths of the set No. 8. a) is the results obtained with the topo-MCL b) with scan-MCL and c) with topo-scan-MCL.

HELSINKI UNIVERSITY OF TECHNOLOGY AUTOMATION TECHNOLOGY RESEARCH REPORTS

No. 19 Xu, B.,
An interactive method for robot control and its application to deburring, November 1998.

No. 20 Zhang, X., Halme A.,
A biofilm reactor for a bacteria fuel cell system, August 1999.

No. 21 Vainio, M.,
Intelligence through interactions – Underwater robot society for distributed operations in closed aquatic environment, October 1999.

No. 22 Appelqvist, P.,
Mechatronics design of a robot society – A case study of minimalist underwater robots for distributed perception and task execution, November 2000.

No. 23 Forsman, P.,
Three-dimensional localization and mapping of static environments by means of mobile perception, November 2001.

No. 24 Selkäinaho, J.,
Adaptive autonomous navigation of mobile robots in unknown environments, December 2002.

No. 25 Oksanen, T.,
Nelijalkainen nelipyöräinen robotti liikkuvana systeeminä, February 2003.

No. 26 Suomela, J.,
From teleoperation to the cognitive human-robot interface, November 2004.

No. 27 Aalto, H.,
Real-time receding horizon optimization of gas pipeline networks, April 2005.

No. 28 Ylönen, S.,
Modularity in service robotics – Techno-economic justification through a case study, December 2006.

No. 29 Appelqvist, A.,
Development of a biocatalytic fuel cell system for low-power electronic applications, December 2006.

No. 30 Leppänen, I.,
Automatic Locomotion Mode Control of Wheel-Legged Robots, September 2007.

No. 31 Oksanen, T.,
Path Planning Algorithms for Agricultural Field Machines, December 2007.

No. 32 Pirttioja, T.,
Applying Agent Technology to Constructing Flexible Monitoring Systems in Process Automation, December 2008.

ISBN 978-951-22-9961-4 (print)
ISBN 978-951-22-9962-1 (pdf)
ISSN 0783-5477